

Data-Driven Face Forensics and Security 2021

Lead Guest Editor: Beijing Chen

Guest Editors: Dengpan Ye, Tanfeng Sun, and Gouenou Coatrieux





Data-Driven Face Forensics and Security 2021

Security and Communication Networks

**Data-Driven Face Forensics and Security
2021**

Lead Guest Editor: Beijing Chen

Guest Editors: Dengpan Ye, Tanfeng Sun, and
Gouenou Coatrieux






Copyright © 2023 Hindawi Limited. All rights reserved.

This is a special issue published in "Security and Communication Networks." All articles are open access articles distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Chief Editor

Roberto Di Pietro, Saudi Arabia

Associate Editors

Jiankun Hu , Australia
Emanuele Maiorana , Italy
David Megias , Spain
Zheng Yan , China

Academic Editors

Saed Saleh Al Rabae , United Arab Emirates
Shadab Alam, Saudi Arabia
Goutham Reddy Alavalapati , USA
Jehad Ali , Republic of Korea
Jehad Ali, Saint Vincent and the Grenadines
Benjamin Aziz , United Kingdom
Taimur Bakhshi , United Kingdom
Spiridon Bakiras , Qatar
Musa Balta, Turkey
Jin Wook Byun , Republic of Korea
Bruno Carpentieri , Italy
Luigi Catuogno , Italy
Ricardo Chaves , Portugal
Chien-Ming Chen , China
Tom Chen , United Kingdom
Stelvio Cimato , Italy
Vincenzo Conti , Italy
Luigi Coppolino , Italy
Salvatore D'Antonio , Italy
Juhriyansyah Dalle, Indonesia
Alfredo De Santis, Italy
Angel M. Del Rey , Spain
Roberto Di Pietro , France
Wenxiu Ding , China
Nicola Dragoni , Denmark
Wei Feng , China
Carmen Fernandez-Gago, Spain
AnMin Fu , China
Clemente Galdi , Italy
Dimitrios Geneiatakis , Italy
Muhammad A. Gondal , Oman
Francesco Gringoli , Italy
Biao Han , China
Jinguang Han , China
Khizar Hayat, Oman
Azeem Irshad, Pakistan

M.A. Jabbar , India
Minho Jo , Republic of Korea
Arijit Karati , Taiwan
ASM Kayes , Australia
Farrukh Aslam Khan , Saudi Arabia
Fazlullah Khan , Pakistan
Kiseon Kim , Republic of Korea
Mehmet Zeki Konyar, Turkey
Sanjeev Kumar, USA
Hyun Kwon, Republic of Korea
Maryline Laurent , France
Jegatha Deborah Lazarus , India
Huaizhi Li , USA
Jiguo Li , China
Xueqin Liang, Finland
Zhe Liu, Canada
Guangchi Liu , USA
Flavio Lombardi , Italy
Yang Lu, China
Vincente Martin, Spain
Weizhi Meng , Denmark
Andrea Michienzi , Italy
Laura Mongioi , Italy
Raul Monroy , Mexico
Naghme Moradpoor , United Kingdom
Leonardo Mostarda , Italy
Mohamed Nassar , Lebanon
Qiang Ni, United Kingdom
Mahmood Niazi , Saudi Arabia
Vincent O. Nyangaresi, Kenya
Lu Ou , China
Hyun-A Park, Republic of Korea
A. Peinado , Spain
Gerardo Pelosi , Italy
Gregorio Martinez Perez , Spain
Pedro Peris-Lopez , Spain
Carla Ràfols, Germany
Francesco Regazzoni, Switzerland
Abdalhossein Rezai , Iran
Helena Rifà-Pous , Spain
Arun Kumar Sangaiah, India
Nadeem Sarwar, Pakistan
Neetesh Saxena, United Kingdom
Savio Sciancalepore , The Netherlands

De Rosal Ignatius Moses Setiadi ,
Indonesia
Wenbo Shi, China
Ghanshyam Singh , South Africa
Vasco Soares, Portugal
Salvatore Sorce , Italy
Abdulhamit Subasi, Saudi Arabia
Zhiyuan Tan , United Kingdom
Keke Tang , China
Je Sen Teh , Australia
Bohui Wang, China
Guojun Wang, China
Jinwei Wang , China
Qichun Wang , China
Hu Xiong , China
Chang Xu , China
Xuehu Yan , China
Anjia Yang , China
Jiachen Yang , China
Yu Yao , China
Yinghui Ye, China
Kuo-Hui Yeh , Taiwan
Yong Yu , China
Xiaohui Yuan , USA
Sherali Zeadally, USA
Leo Y. Zhang, Australia
Tao Zhang, China
Youwen Zhu , China
Zhengyu Zhu , China

Contents

Local Corner and Motion Key Point Trajectory Extraction for Facial Forgery Identification

Qingtong Liu  and Ziyu Xue 




Research Article (10 pages), Article ID 1648912, Volume 2023 (2023)

Manipulated Faces Detection with Adaptive Filter

Chaoyang Peng , Tanfeng Sun , Zhongjie Mi, and Lihong Yao


Research Article (8 pages), Article ID 8609661, Volume 2022 (2022)

Multisemantic Path Neural Network for Deepfake Detection

Nan Wu , Xin Jin , Qian Jiang , Puming Wang , Ya Zhang , Shaowen Yao , and Wei Zhou 



Research Article (14 pages), Article ID 4976848, Volume 2022 (2022)

DRT-Unet: A Segmentation Network for Aiding Brain Tumor Diagnosis

Shujing Li and Linguo Li 



Research Article (11 pages), Article ID 2546466, Volume 2022 (2022)

Embedding Guided End-to-End Framework for Robust Image Watermarking

Beibei Zhang , Yunqing Wu, and Beijing Chen 





Research Article (11 pages), Article ID 7259469, Volume 2022 (2022)

Semantic Modeling and Pixel Discrimination for Image Manipulation Detection

Ziyu Xue , Xiuhua Jiang, and Qingtong Liu 


Research Article (10 pages), Article ID 9755509, Volume 2022 (2022)

Domain Transferred Image Recognition via Generative Adversarial Network

Haoqi Hu , Sheng Li , Zhenxing Qian , and Xinpeng Zhang 






Research Article (11 pages), Article ID 4466426, Volume 2022 (2022)

Integrating Temporal and Spatial Attention for Video Action Recognition

Yuanding Zhou, Baopu Li, Zhihui Wang , and Haojie Li

Research Article (8 pages), Article ID 5094801, Volume 2022 (2022)

Separable Reversible Data Hiding in Encrypted VQ-Encoded Images

Fang Cao , Yujie Fu, Heng Yao , Mian Zou , Jian Li , and Chuan Qin 





Research Article (16 pages), Article ID 1227926, Volume 2022 (2022)

An Improved Self-Training Model with Fine-Tuning Teacher/Student Exchange Strategy to Detect Computer-Generated Images

Ye Yao , Shuhui Liu, Hui Wang , Zhangyi Shen , and Xuan Ni


Research Article (14 pages), Article ID 3493336, Volume 2022 (2022)

Coverless Video Steganography Based on Audio and Frame Features

Chunhu Zhang , Yun Tan , Jiaohua Qin , and Xuyu Xiang 



Research Article (14 pages), Article ID 1154098, Volume 2022 (2022)

Detection of Face Morphing Attacks Based on Patch-Level Features and Lightweight Networks

Min Long , Xuan Zhao, Le-Bing Zhang, and Fei Peng







Research Article (12 pages), Article ID 7460330, Volume 2022 (2022)

Toward Steganographic Payload Location via Neighboring Weight Algorithm

Tong Qiao , Xiangyang Luo , Binmin Pan, Yuxing Chen, and Xiaoshuai Wu





Research Article (17 pages), Article ID 1400708, Volume 2022 (2022)

Face Forgery Detection Based on the Improved Siamese Network

Bo Wang , Yucai Li , Xiaohan Wu , Yanyan Ma , Zengren Song , and Mingkan Wu 

Research Article (13 pages), Article ID 5169873, Volume 2022 (2022)

Perceptual Image Hashing Based on Multitask Neural Network

Cheng Xiong , Enli Liu, Xinran Li , Heng Yao , Lei Zhang, and Chuan Qin 

Research Article (11 pages), Article ID 8297244, Volume 2021 (2021)

Research Article

Local Corner and Motion Key Point Trajectory Extraction for Facial Forgery Identification

Qingtong Liu  and Ziyu Xue 

Academy of Broadcasting Science, NRTA, Beijing, China

Correspondence should be addressed to Qingtong Liu; qingtong_liu@126.com

Received 24 February 2022; Revised 2 August 2022; Accepted 11 April 2023; Published 15 May 2023

Academic Editor: Beijing Chen

Copyright © 2023 Qingtong Liu and Ziyu Xue. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

At present, the development of deep forgery technology has brought new challenges to media content forensics, and the use of deep forgery identification methods to identify forged audio and video has become a significant focus of research and difficulty. Deep forgery technology and forensic technology play a mutual game and promote each other's development. This paper proposes a spatiotemporal local feature abstraction (STLFA) framework for facial forgery identification to solve the media industry challenges of deep forgery technology. To adequately utilize local facial features, we combine facial key points, key point movement, and facial corner points to detect forgery content. This paper establishes a spatiotemporal relation, which realizes face forgery detection by identifying abnormalities of facial keypoints and corner points for interframe judgments. Meanwhile, we utilize RNNs to predict the sequences from facial key point movement abnormalities and corner points for interframe. Experimental results show that our method achieves better performance than some existing methods and good anticompression forgery face detection performance on FF++.

1. Introduction

Media content forgery has brought some security problems to society. Especially with the development of autoencoders (AEs) [1] and generative adversarial networks (GANs) [2], media content forgery has become easy to achieve through deep forgery techniques. The techniques usually utilize deep learning methods to alter a person's identity in a video to synthesize a piece of media content that does not exist. Deep forgery identification techniques include both image-level detection and video-level detection.

Forgery detection of images or video frames is mostly the detection of forged video content, including color inconsistencies and semantic inconsistencies. Image forgery detection can be divided into detecting the image as a whole and detecting the facial area, according to the detection dimension. Forgery detection of the image as a whole is mainly to detect the physical properties of the image, such as the direction of the image's light source [3], the saturated pixel frequency [4], and the spectral sensitivity [4]. It is

classified by judging the difference between forged and authentic images. Forgery detection for facial regions includes inconsistent iris color, missing tooth gaps, and inconsistent eye reflexes, including detection of facial artifacts using light estimation, global consistency and geometric estimation [5], corneal highlight region consistency detection [6], and facial artifact detection [7].

The detection of video sequences is mainly performed by combining optical flow anomalies, motion incoherence, or anomalies between video frames. Forgery detection based on optical flow mainly calculates the optical flow field of the target in the video and detects the inconsistency of the optical flow field [8]. Some authors utilize eye blinks [9], abnormal head movements [10], and facial distortions [11] to detect incoherent motion or abnormal behaviors in consecutive frames.

However, the early works were mainly focused on global features. Specifically, we notice that forgery detection features are particularly evident in key facial organs such as the eyes, nose, and mouth [5, 6, 12]. For example, Xue et al. [12]

found that only using facial organs such as the nose, lips, eyes, eyebrows, and chin can detect deep forgery very well.

Based on this, we first consider constructing the facial organs' relation. These organs can be abstracted to local features and represented by sequential vectors. We then adopt recurrent neural networks (RNNs) to capture their internal properties or differences to obtain instructive guidance that describes whether the face is falsified. For comprehensive detection, we realize face forgery detection for key facial local regions such as the lips, eyes, nose, eyebrows, and chin, thus achieving impressive performance. The contributions of our work are summarized as follows:

- (1) We propose a spatiotemporal local feature abstraction (STLFA) framework for facial forgery identification, which establishes local features' relation via an organ-specific method.
- (2) In STLFA, we combine abnormal facial movement detection and facial landmark time discontinuity detection to analyze the facial key point and corner point features frame by frame. Meanwhile, we judge video sequences' key point movement and corner point number transformation to achieve forgery identification of images and videos.
- (3) This paper demonstrates the effectiveness and robustness of the proposed method and discusses and analyzes the advantages and disadvantages of STLFA.

2. Related Works

2.1. Deep Forgery Discrimination Based on Image or Video Frames. Currently, most forgery detection of images or video frames is performed by detecting manual features for forgery identification. The detection subject can be divided into two categories: image detection and inconsistency detection only for human faces.

Image forgery detection mainly detects the inconsistent lighting conditions and color inconsistencies in images. Chen et al. [13] proposed a robust dual-stream network by integrating dual-color spaces RGB and YCbCr using an improved Xception model, which considers both the luminance and chrominance components of dual-color spaces (RGB and YCbCr) to enhance the robustness. Johnson and Farid [3] proposed a method to detect lighting inconsistencies by estimating the direction of point light sources in a single image to estimate the consistency of light sources for the whole image. McCloskey and Albright [4] analyzed the structure of the popular GAN network. They found that the image generated by the GAN network differs from the captured image in color processing. They propose a method for forgery classification by saturated pixel frequency detection and spectral sensitivity detection.

The forgery detection of inconsistencies in the person's face focuses on the incomplete consideration of semantics in the content generation process by the deep forgery method, resulting in the generation of a person with inconsistent iris colors in the left and right eyes, inconsistent reflections, and uneven gaps in the teeth. Matern et al. [5] detected facial

artifacts based on detecting intraframe image artifacts using light estimation, global consistency, and geometric estimation. Hu et al. [6] proposed a scheme to study whether the highlight patterns on the corneas of two eyes are consistent to determine whether they are fake. Li and Lyu [7] determined the forgery traces by detecting artifacts traced from the affine transformation during face forgery.

In order to integrate the features of facial regions, some authors proposed novel approaches. Wang et al. [14] proposed a method that fused facial region feature descriptor for forgery determination by extracting feature points of a person's face. Xue et al. [12] built a transformer model for a deepfake-detection method by organs to obtain the deepfake features. Yang et al. [15] proposed a method for detecting differences in face textures by amplifying the texture differences between genuine and fake images and using a bootstrap filter to enhance postprocessing-induced texture artifacts and display the underlying features of the artifacts.

2.2. Deep Forgery Discrimination Based on Video Sequences.

The video sequence-based deep forgery approaches have more detection items than the image-based deep forgery approach. The forged video generation process is frame-by-frame leading to optical flow inconsistencies between the preceding and following frames and motion anomalies.

In terms of forgery identification based on optical flow detection, Amerini et al. [8] proposed a forgery detection method based on optical flow anomalies between different frames by extracting the correlation of the optical flow field and using a CNN classifier for classification. Trinh et al. [16] proposed a forgery detection framework by superimposing optical flow fields on RGB images for forgery detection. Caldelli et al. [17] proposed a CNN-based classification method to distinguish motion dissimilarities in the temporal structure of video sequences by using optical flow fields.

In terms of forgery identification based on abnormal motion detection, Li et al. [9] proposed a GAN-based model that could not represent blinking in fake synthetic videos, enabling the detection of blink inconsistencies. Yang et al. [10] proposed a detection method based on the inconsistency of 3D head pose estimation by extracting the coordinates of facial key points and calculating the direction vector difference between the center of the face and the coordinates of peripheral key points to achieve deep forgery detection. Sun et al. [11] proposed a geometric feature calibration module to determine the accuracy of interframe geometric features to determine the abnormal facial movements of characters.

3. Methods

3.1. Framework. In this section, we provide a detailed illustration of our proposed method. Figure 1 illustrates the architecture of STLFA. We used facial preprocessing modules to crop the eight facial organ regions, including the left eyebrow, right eyebrow, left eye, right eye, nose, mouth, inner mouth, and chin. We built a sequence group by facial

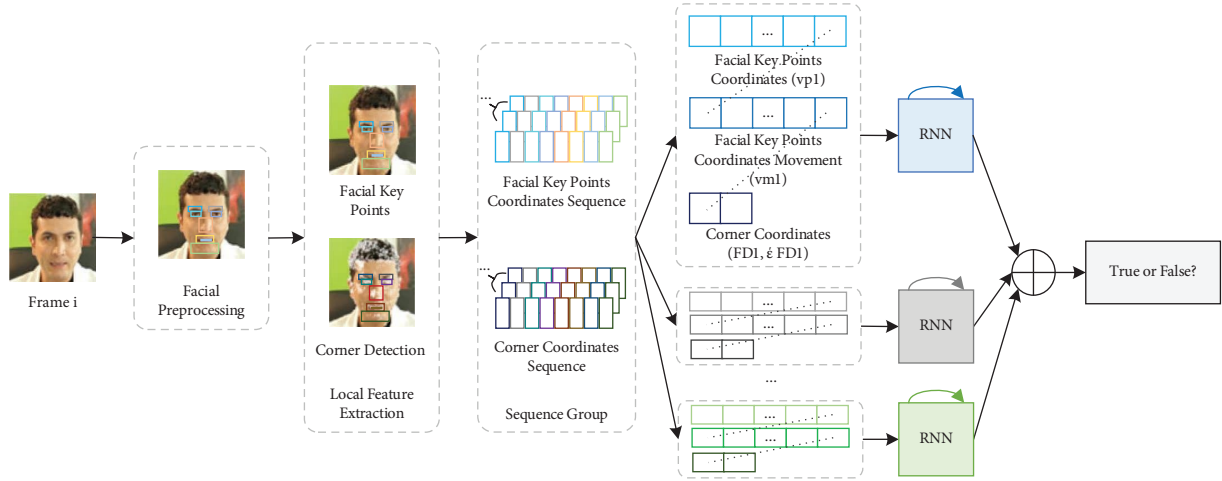


FIGURE 1: The framework of STLFA. The face image in this figure comes from the FF++ dataset [18] obtained from open access.

key points, key point movement, number of corner points, and number of variations. Meanwhile, RNN models are trained for each region until they have the detection ability. After that, we integrate the results from the RNNs and obtain the final prediction.

3.2. Facial Preprocessing. The facial preprocessing module mainly contains three steps: face detection, face landmark detection, and landmark alignment. Following [11], we use tracking and denoising methods to match the key points between video sequences to obtain the complete facial key point coordinates and coordinate movement. We utilized the Lucas–Kanade (LK) operation in the tracking method to track the coordinate points and forward-backward processes to eliminate inaccurate predictions. Meanwhile, the denoising method is used to solve the noise caused by the LK operation and to ensure the stability of the landmark, using the Kalman filter to integrate the prediction information.

3.3. Facial Key Points Extraction

3.3.1. Facial Key Points Coordinates Extraction. The facial key point coordinate detection method requires cropping the preprocessed image. After that, we detect 68 facial key points representing the facial shape, as shown in Figure 2(a). We select the key point frame to extract eight facial key organ regions based on the 68 key points, as demonstrated in Figure 2(b). We create vector v_p for each key organ region.

$$v_p = [v_{p_1}, v_{p_2}, \dots, v_{p_8}]. \quad (1)$$

Each region can be expressed as v_{p_i} :

$$v_{p_i} = [x_i^1, y_i^1, x_i^2, y_i^2, \dots, x_i^n, y_i^n], \quad (2)$$

where x_i^1 is the horizontal coordinate of the first key point in region i and y_i^1 is the vertical coordinate of region i .

3.3.2. Corner Extraction

(1) Motivation for Using FAST Feature Points. The FAST algorithm is a corner detection algorithm mainly used to extract the feature points in the image. Based on the feature point information, the translation, distortion, and rotation objects in the dynamic process are associated with realizing the target tracking in a series of images of dynamic imaging and positioning. Wang et al. [14] found that although the fake video face was highly similar to the original video face, it still lost many fine details used to determine the FAST feature points and found that the phenomenon was more evident in the local area of the face. Based on this observation, we design a FAST feature descriptor to extract the phenomenon of the occasional failure of face-changing in the local area of the fake video and further complete the face forgery detection.

(2) Extraction Algorithm Feature Point of FAST. Features from accelerated segment test (FAST) [19] is an efficient corner point detection method mainly used for feature extraction of image corner points. The FAST method builds up the intensity of a pixel point I_p , sets the threshold value to t , and creates a Bresenham circle for 16-pixel points around p , as shown in Figure 3(a).

Designating pixel point p as a corner point if there is a set of n consecutive pixels in the circle that are all brighter than $I_p + t$ or darker than $I_p - t$.

In order to speed up the operation, the pixel points compared with I_p can be simplified and set to 1, 5, 9, and 13, as shown in Figure 3(b). This paper focuses on establishing FAST corner point detection for eight regions extracted, such as the eyes, nose, lips, and eyebrows, and establishing corner point comparisons between frames, as shown in Figure 3(c).

We define pixel p as a corner when the circle in Figure 3(a) has a group of consecutive pixel points. Meanwhile,

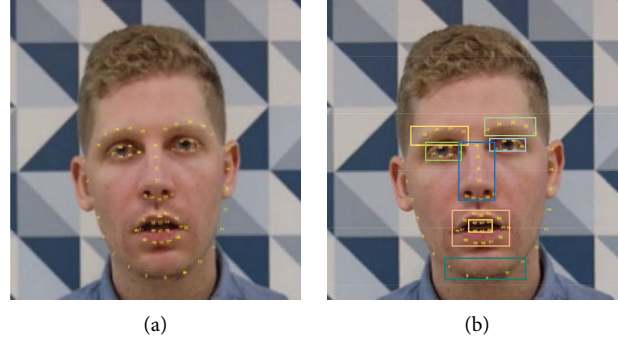


FIGURE 2: Facial key point coordinate detection: (a) 68 key points of facial contour. The face image is from FF++ [18]. (b) The key region cropped. The face image is from FF++ [18].

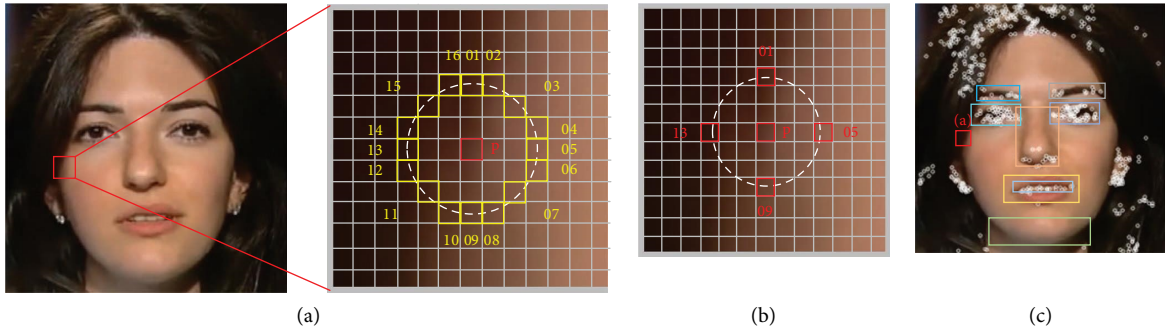


FIGURE 3: FAST feature point extraction algorithm: (a) p is the selected corner point, and a Bresenham circle is established around the point p . The face image is from FF++ [18]. (b) Simplified corner operations. The face image is from FF++ [18]. (c) Fast corner points detection in eight regions. The face image is from FF++ [18].

the points are brighter than $I_p + t$ or darker than $I_p - t$. In order to speed up the operation, the points can be simplified and only use points 1, 5, 9, and 13 to calculate, as shown in Figure 3(b). We focus on establishing FAST corner point detection for eight regions, as shown in Figure 3(c), and setting corner point comparisons between frames.

3.4. Abnormal Facial Movement Detection

3.4.1. Facial Shape Movement Abnormal Detection. Facial shape movement detection is based on the extraction of 68 feature points of facial feature extraction; the facial area is divided into 8 areas, and the temporal movement pattern of the feature points in each area is established for each area to realize facial shape movement abnormal detection. We analyze the movement of key points in each region and build a key points coordinate vector $v_{l_k}^i$.

$$v_{l_k}^i = [x_1^i, y_1^i, x_2^i, y_2^i, \dots, x_{68}^i, y_{68}^i]. \quad (3)$$

The key point coordinate vector of the eight regions collection in frame i can be expressed as v_l^i :

$$v_l^i = [v_{l_1}^i, v_{l_2}^i, \dots, v_{l_8}^i], \quad (4)$$

where $v_{l_1}^i \sim v_{l_8}^i$ represents the respective vectors of the eight regions in frame i and the corresponding key points are as

follows: 6~10 represent the chin, 17~21 points represent the left eyebrow, 22~26 represent the right eyebrow, 36~41 represent the left eye, 42~47 represent the right eye, 27~35 represent the nose, 48~60 represent the mouth, and 61~67 stands for the inner mouth.

Then, we use $v_{l_1}^i \sim v_{l_8}^i$, extracted frame by frame, to provide clues for subsequent temporal discontinuity detection of facial motion morphology.

3.4.2. Facial Corner Abnormal Detection. Following [14], we use FAST to obtain feature points with a descriptor des of 32 dimensions. We assume that the number of corner points FD_k^i of the focus organ region k is in frame i , then FD_k^i can be expressed as follows:

$$FD_k^i = \sum_{k_i} \text{des}[x], x \in [1, 32]. \quad (5)$$

In this way, a feature vector FD^i can be created for the eight regions:

$$FD^i = [FD_1^i, FD_2^i, \dots, FD_8^i], \quad (6)$$

where FD_k^i is a statistical vector based on corner points in region i , containing the number of corner points in region k at frame i . We create time series based on FD_k^i to detect clues of alternating authentic and forgery faces in forgery videos.

3.5. Facial Landmark Time Discontinuity Detection

3.5.1. Facial Key Points Time Discontinuity Detection.

We detect the temporal discontinuity of facial key point displacement between frames based on the displacement information of facial key points between consecutive frames. We analyze the movement of key points in each region and build a key point coordinate movement vector $v_{m_k}^i$; each region can be expressed as follows:

$$v_{m_k}^i = [\Delta x_i^1, \Delta y_i^1, \Delta x_i^2, \Delta y_i^2, \dots, \Delta x_i^n, \Delta y_i^n]. \quad (7)$$

The key point coordinate movement vector of the eight regions collection in frame i can be expressed as v_m^i :

$$v_m^i = [v_{m_1}^i, v_{m_2}^i, \dots, v_{m_8}^i], \quad (8)$$

where Δx_i^1 is the adjacent frames variation in the horizontal coordinates; we can calculate Δx_i^1 using $|x_{i+1}^1 - x_i^1|$, the same as Δy_i^1 .

3.5.2. *FAST Feature Time Discontinuity Detection.* The FD_k^i in Section 3.4.2 is the corner number vector of the described local region, and we use this vector to build the corner number difference vector ΔFD_k^i between consecutive frames:

$$\Delta FD_k^i = [FD_k^2 - FD_k^1, FD_k^3 - FD_k^2, \dots, FD_k^i - FD_k^{i-1}]. \quad (9)$$

ΔFD_k^i is the difference between the number of corners in region k in frame i and the number in region k in frame $i - 1$. The statistical vector ΔFD^i of the difference in the number of the corners in the whole facial region can be expressed as follows:

$$\Delta FD^i = [FD_1^i, FD_2^i, \dots, FD_8^i]. \quad (10)$$

We use ΔFD^i to detect nonsmooth facial corner number changes in the video.

3.6. Facial Forgery Prediction

3.6.1. *Facial Feature Vector Association.* Based on $v_{l_r}^i$, $v_{m_k}^i$, FD_k^i , and ΔFD_k^i obtained in Sections 3.4 and 3.5, the local facial feature fusion vector $v_{f_k}^i$ is formed by concatenating the four types of feature vector sequences:

$$v_{f_k}^i = [v_{l_r}^i, v_{m_k}^i, FD_k^i, \Delta FD_k^i]. \quad (11)$$

Then, the local facial feature fusion vector for region k of the entire video can be expressed as follows:

$$v_{f_k} = [v_{f_k}^1, v_{f_k}^2, \dots, v_{f_k}^n]. \quad (12)$$

We utilize a series of the local facial feature fusion vectors $v_{f_1} \sim v_{f_8}$ to represent the facial fusion features. After that, we use the connected feature vector v_{f_k} to train a dual-stream RNN model for each of the eight regions to classify the forgery videos.

3.6.2. *RNN-Based Deep Forgery Detection.* We utilize RNNs to model local facial feature sequences. In order to ensure an identical input dimension of the RNN and to achieve deep forgery detection at the video level, each video sample used

as input is cut into a fixed length, and a fixed number of key frames are extracted. Based on the extraction results, the RNN parameters are selected for training to achieve deep forgery detection of the overall video.

Through the embedding process, the RNNs are adopted to model the feature sequences of each local region, learning the shape movement pattern, landmark difference pattern, and FAST feature point variation pattern. Then, the fully connected (FC) network is connected to each RNN output layer. Furthermore, calculate 8 FC layers output average result as the final prediction to achieve deep forgery detection based on the local regions of the face. We utilize F to represent this process:

$$F(R_1(v_{f_1}), R_2(v_{f_2}), \dots, R_8(v_{f_8})). \quad (13)$$

4. Experiments

4.1. Datasets

- (1) FaceForensics++ (FF++) [18]: FF++ is one of the benchmark datasets for large-scale deep forgery detection, with a total of over 1,000 segments, more than 1.5 million frames in total, and over 1.5 TB of video data in the original video format. Meanwhile, a face detector is used to filter the video footage to ensure that there are three video qualities in the FF++ dataset, Raw, c23, and c40, characterized by many forged video segments, and a variety of deep forgery methods are considered.
- (2) Celeb-DF [20]: The Celeb-DF (v2) dataset is a large-scale deepfake forensic dataset that addresses the shortcomings of poor forged video quality, apparent forgery traces, and flickering video faces. The Celeb-DF (v2) dataset improves the deep forgery generation method and the face key point localization method to obtain stable fake video content quality. The dataset contains 590 raw videos collected from YouTube with categories of different ages, races, and genders. 5639 HD deepfake videos are the same quality as the online broadcast videos.
- (3) DFDC preview dataset [21]: This dataset comes from The Deepfake Detection Challenge hosted by Facebook. It is the preliminary dataset for the competition. It consists of 5,214 videos, of which the ratio of true and false content is 1:0.28, and forgery data contain data generated by two deep forgery methods. Each video is a clip of about 15 s.

4.2. *Experiment Settings.* During preprocessing, DLIB was used for face cropping and face landmark detection, and FAST detector and BRIEF descriptors were used for corner point detection and description. In the classification process, a bidirectional recurrent neural network connects to the feature sequences in the respective regions. Each RNN in the detection framework consists of a GRU (gated recurrent unit) with a hidden layer feature output dimension of 64. A dropout layer is set between the input and the RNN, using a fully connected network to connect to the output of the

RNN layer. Using two $dr = 0.5$ dropout layers separated between the RNN layer and the fully connected layer and inside, these experimental parameter settings partly refer to existing research results [22].

In the experimental dataset section, the ratio of training data to test data was 7 : 3, with 120 frames drawn from each video. The model was optimized using the Adam optimizer for the specific training process. We initialize the learning rate at 0.005, set the batch size to 1024, and the maximum number of iterations Epoch was 800 rounds. The experiments in this paper use AUC (area under curve) to evaluate the performance of the deep forgery detection model, and the AUC is calculated as follows:

$$AUC = \frac{\sum \text{pred}_{\text{pos}} > \text{pred}_{\text{neg}}}{\text{positiveNum} * \text{negativeNum}}, \quad (14)$$

where pred_{pos} is the predicted probability of getting a positive sample, pred_{neg} is the predicted probability of getting a negative sample, positiveNum is the number of positive samples, negativeNum is the number of negative samples, and AUC is the number of samples where the predicted probability of a positive sample is greater than the predicted probability of a negative sample in the $\text{positiveNum} * \text{negativeNum}$ sample.

4.3. Experiments

4.3.1. Partial Organ Comparison. In this paper, experiments are conducted on the FF++ dataset to compare each organ region module’s detection effect to verify each organ’s region detection effect on deep forgery. In this paper, following the idea of [14], eight key regions such as the left eyebrow, right eyebrow, left eye, right eye, nose, mouth, inner mouth, and chin were set up and compared, as shown in Table 1. The “Points” results are obtained using facial key point coordinate detection and facial key point coordinate movement detection, “Coordinate” indicates the detection result using only the facial key point coordinates, and “Movement” indicates the detection result using only the facial key point movement coordinates. “C+M” indicates the result obtained by combining the key point coordinate detection and the facial key point coordinate movement detection. “Corners” is the result obtained using FAST corner number detection and corner number change detection. “All” means that the results of “Points” and “Corners” are combined with the experimental results of FAST features, and the RNNs of each segment are trained separately.

From Table 1, all local organs can be used individually in the FF++ dataset to detect whether the images contain forgeries. This paper observes that among the eight organ regions, the eyebrows, eyes, and mouth have the highest accuracy rate, while the nose and chin have a low accuracy rate. Also, in the “Points” detection group, where three experiments were set up, it was seen that “Coordinate” could perform a single-frame detection task with an average detection rate of 87.2%. “Movement” is the detection method combined with video sequences, with an average detection

rate of 82.6%. The combination of “Coordinate” and “Movement” enables the combination of abnormal facial movement detection and facial landmark time discontinuity detection, allowing for more effective acquisition of key facial features with an accuracy rate of 91.1%.

4.3.2. Ablation Study. In this paper, we use the frame-level AUC to verify the effectiveness of face key point and corner point detection on deep forgery detection, respectively, to validate the proposed method. The models in the experiments are trained on FF++ (raw) and tested on three datasets: FF++, DFDC Preview, and Celeb-DF. The results are shown in Table 2.

The experimental results show that “Points” and “Corners” have similar detection results in terms of AUC, with an average of 71.3% and 74.1%, respectively, and all the best detection was achieved by “All,” with an AUC of 75.9%. Meanwhile, in the FF++, DFDC Preview, and Celeb-DF datasets, the AUC values of “All” were higher than those of “Points” and “Corners” and “All” has a higher AUC than “Points” and “Corners.” This proves that the method proposed in this paper, which combines facial key point and corner point detection, is reasonable and effective.

4.3.3. Comparison Experiments. In this paper, using frame-level AUC evaluation, we selected mainstream deep forgery detection methods based on full-frame face region forgery detection [18], fake face edge fusion region detection [23], facial landmark feature enhancement forgery and detection [11], visual distortion detection [24], and capsule network forgery detection [25]. Tests were carried out on datasets such as FF++, DFDC Preview, and Celeb-DF. We refer to the detection results of [11, 14], as shown in Table 3. In the FF++ dataset, “raw” represents the uncompressed data and “c40” represents the compressed LQ data.

As can be seen from Table 3, the AUC results of the proposed method on FF++ are better than those of mainstream methods such as Xception [18], Face X-ray [23], LRNet [11], DSP-FWA [24], and Capsule [25]. In particular, in the experimental group of “c40,” the proposed method has better robustness for low-quality forged video identification, with a 1.7% improvement over LRNet [11] and a 35.8% improvement over Face X-ray [23].

In anticompression forgery face detection, our work shows a good forgery face detection performance. The method in this paper extracts the geometric features of the local facial region by combining the local facial key points and the corner. The extracted features have more robust and lower cost characteristics and have high sensitivity in detecting changes in the number of the corner. The strategy designed in this paper for face forgery detection through 8 local facial regions improves the accuracy of overall face forgery detection by reducing the detection error of a single region. The effectiveness of our strategy is also verified on FF++ (Raw, c40).

The low-complexity and high-performance geometric feature extraction method designed in this paper can effectively reduce the impact of image compression on the face

TABLE 1: Comparison table of local organs (Acc (%)).

Region attribute	Left eyebrow	Right eyebrow	Left eye	Right eye	Nose	Mouth	Inner mouth	Chin	Avg	
Points	Coordinate	92.5	90.8	90.6	91.5	85.1	88.2	74.7	83.8	87.2
	Movement	83.4	82.7	84.5	83.2	80.2	84.3	82.4	79.8	82.6
	C + M	93.4	91.3	91.6	92.7	87.2	90.1	94.7	87.8	91.1
Corners		94.2	92.1	92.3	92.5	84.7	93.4	88.6	83.4	90.2
All		97.2	97.7	98.8	98.3	96.4	98.6	95.1	94.3	97.0

The bold values are used to highlight the results of the experiments conducted for this study. Specifically, they represent the performance of the proposed method in each experimental group.

TABLE 2: Ablation experiments (AUC (%)).

Datasets	FF++ (6284)	DFDC Preview (5214)	Celeb-DF (6819)	Avg
Points	99.2	56.2	58.4	71.8
Corners	96.7	62.7	63.1	74.5
All	99.9	63.5	64.3	76.3

The bold values are used to indicate the experimental records where the proposed method, discussed in this paper, demonstrated the most favorable outcomes within each experimental group. By highlighting these values in bold, we aim to emphasize the superior performance achieved by our method in those specific experimental conditions.

TABLE 3: AUC (%) results of the proposed method and mainstream methods on the FF++ dataset.

Methods	FF++	
	Raw	c40
Xception [18]	99.7	86.5
Face X-ray [23]	99.1	61.6
LRNet [11]	99.9	95.7
DSP-FWA [24]	93.0	—
Capsule [25]	96.6	—
Single XceptionNet [26]	—	97.8
Chen et al. [27]	99.92	95.2
SPSL [28]	—	82.8
PCL + I2G [29]	99.79	—
FTCN [30]	99.7	—
Lip forensics [31]	98.9	94.2
FDL [32]	99.7	92.4
Ours	99.9	97.6

The bold values are used to highlight the experimental records that represent the most optimal performance within each experimental group. Specifically, the bold values labeled as “Ours” indicate the results obtained from the experiments conducted using our proposed method.

forgery detection task, and the experimental results further demonstrate this. We compared this method’s training and testing results and other methods on the FF++ (Raw, c40) dataset in Table 3. The results show that our method achieves better performance than some existing methods, with a difference of 0.4% in AUC compared to the Single XceptionNet [26] method on FF++ (c40), and has better anticompensation forgery face detection performance. The detection performance suffers less interference on c40 data.

4.3.4. Cross-Dataset Experiments. Our method can tolerate the local area detection, such as eyes, nose, and other organs, which is suitable for detecting the forgery videos with stain and shelter. To further demonstrate the robustness of our method, the models trained on FF++ (raw) were selected and tested on the DFDC Preview and Celeb-DF datasets. The results of training and testing on FF++ (raw, c40) in Table 4 sets cross-dataset experiments in individual organs and organ combinations.

The experimental results show that our method is innovative and can only use individual organs to detect forgery videos with defilement and stain. Meanwhile, using all organ regions has higher average accuracy. To further verify the ability of our method, we set up cross-dataset experiments to compare with the state-of-arts in Table 5.

The test results are shown in Table 5, Xception [18], LRNet [11], DSP-FWA [24], Capsule [25], Single XceptionNet [26], FWA [7], LipForensics [31], STIL [33], ADDNet-3D [34], and ours are compared. The method has certain advantages in the existing DFDC Preview cross-dataset test results, but the effect still needs to be further improved in the cross-dataset test results. The specific reasons are analyzed as follows: the framework of this paper utilizes the spatial and temporal features such as the spatial position of facial feature points and the statistical number of FAST corner points and shows good performance on the FF++ dataset. This paper strengthens the description and distinguishing capabilities of forgery faces to a certain extent by using geometric features and uses the RNN to model the

TABLE 4: The detection accuracy in cross-dataset experiments only uses local organs and organ combinations (Acc (%)).

Region attribute		FF++	CelebDF	DFDC Preview
Single attribute	Left eyebrow	97.2	63.9	61.2
	Right eyebrow	97.7	64.3	63.1
	Left eye	98.8	66.7	67.8
	Right eye	98.3	66.2	63.7
	Nose	96.4	61.8	60.3
	Mouth	98.6	66.4	64.1
	Inner mouth	95.1	59.3	57.9
	Chin	94.3	58.7	55.6
Multiattribute	Eyes	98.9	64.2	62.7
	Eyes + eyebrows	99.1	65.1	63.1
	Eyes + mouth	99.2	64.8	62.9
	Mouth + inner mouth	97.4	63.2	62.2
	Nose + mouth + inner mouth	97.8	63.9	62.4
	Mouth + inner mouth + chin	96.1	60.1	59.8
All		99.4	65.8	63.7

TABLE 5: Cross-dataset experiments (AUC (%)).

Methods	Celeb-DF	DFDC Preview
<i>Train on FF++ (raw)</i>		
Xception [18]	48.2	49.9
LRNet [11]	56.9	—
DSP-FWA [24]	64.6	—
Capsule [25]	57.5	53.3
FWA [7]	56.9	—
Ours (raw)	64.8	63.5
<i>Train on FF++ (c23)</i>		
FWA [7]	53.9	—
LipForensics [31]	82.4	—
Ours (c23)	65.1	64.1
<i>Train on FF++ (c40)</i>		
STIL [33]	75.58	—
ADDNet-3D [34]	60.85	—
Ours (c40)	64.7	63.8

The bold values are used to highlight the experimental records that represent the most optimal performance within each experimental group. Specifically, the bold values labeled as ‘‘Ours’’ indicate the results obtained from the experiments conducted using our proposed method.

time series of features to complete fake face detection, which verifies the effectiveness of the framework. Applying geometric features improves the sensitivity to detecting facial feature point motion patterns and differential changes to a certain extent. Still, in the face of forging changes in the scene around the face of different datasets, the feature extraction method in this framework needs to be further optimized. Obtaining more effective forgery face features is the further optimization direction of this framework.

4.4. Discussion. Although the proposed method utilizes RNNs to model local facial feature sequences, it achieves deepfake discrimination through abnormal facial movement detection and facial landmark time discontinuity detection and exhibits good detection performance and compression resistance. Our method mainly mines the detection performance of each local face region for deep forgery and can effectively learn and model

local face regions’ forgery features and patterns. However, since the sample distribution of the FF++ dataset cannot represent all deep forgery techniques, the generalization of this method under the new data distribution is not explicitly guaranteed, which may lead to the degradation of performance in cross-database testing. Research on the generalization problem will be our future goal.

5. Conclusion

The development of deep forgery technology has brought new challenges to the authenticity of media content. The mutual promotion of deep forgery technology and forensics technology is prominent in addressing the challenges brought by deep forgery technology to the media industry. We focus on the consistency of facial key points and corner points’ coordinates and propose a spatiotemporal local feature abstraction (STLFA) framework for facial forgery identification, which establishes local features’ relation via an organ-specific method, which combines abnormal facial movement detection and facial landmark time discontinuity detection to analyze the facial key point, and corner point features frame by frame. It is mainly to detect the consistency of the movement of facial key point coordinates and the facial corner point number variations. At the same time, the method utilizes the bidirectional RNN to establish the sequence in eight local facial regions to model the facial shape pattern, the key point movement pattern, and the corner point number variations.

Experimental results show that our method performs better than some existing methods and achieves good anticompensation forgery face detection performance on FF++. At the same time, for the detection of face forgery, the generalization ability under cross-dataset testing is also important. Therefore, a robust method with strong generalization ability is the goal of our future work.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was supported by the National Fiscal Expenditure Program of China under grant 130016000000200003.

References

- [1] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," 2013, <https://arxiv.org/abs/1312.6114>.
- [2] I. Goodfellow, J. Pouget-Abadie, and M. Mirza, "Generative adversarial nets," *Advances in Neural Information Processing Systems*, vol. 27, 2014.
- [3] M. K. Johnson and H. Farid, "Exposing digital forgeries by detecting inconsistencies in lighting," in *Proceedings of the 7th workshop on Multimedia and security*, pp. 1–10, New York, NY, USA, August 2005.
- [4] S. McCloskey and M. Albricht, "Detecting gan-generated imagery using color cues," 2018, <https://arxiv.org/abs/1812.08247>.
- [5] F. Matern, C. Riess, and M. Stamminger, "Exploiting visual artifacts to expose deepfakes and face manipulations," in *Proceedings of the 2019 IEEE Winter Applications of Computer Vision Workshops (WACVW)*, pp. 83–92, IEEE, Waikoloa, HI, USA, January 2019.
- [6] S. Hu, Y. Li, and S. Lyu, "Exposing GAN-generated faces using inconsistent corneal specular highlights," in *Proceedings of the ICASSP 2021-2021 IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, pp. 2500–2504, IEEE, Toronto, Canada, June 2021.
- [7] Y. Li and S. Lyu, "Exposing deepfake videos by detecting face warping artifacts," 2018, <https://arxiv.org/abs/1811.00656>.
- [8] I. Amerini, L. Galteri, and R. Caldelli, "Deepfake video detection through optical flow based cnn," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, Seoul, Korea (South), October 2019.
- [9] Y. Li, M. C. Chang, and S. Lyu, "In ictu oculi: exposing ai generated fake face videos by detecting eye blinking," 2018, <https://arxiv.org/abs/1806.02877>.
- [10] X. Yang, Y. Li, and S. Lyu, "Exposing deep fakes using inconsistent head poses," in *Proceedings of the ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8261–8265, IEEE, Brighton, UK, May 2019.
- [11] Z. Sun, Y. Han, and Z. Hua, "Improving the efficiency and robustness of deepfakes detection through precise geometric features," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3609–3618, New Orleans, LA, USA, June 2021.
- [12] Z. Xue, Q. Liu, H. Shi, R. Zou, and X. Jiang, "A transformer-based DeepFake-detection method for facial organs," *Electronics*, vol. 11, no. 24, p. 4143, 2022.
- [13] B. Chen, X. Liu, Y. Zheng, G. Zhao, and Y. Q. Shi, "A robust GAN-generated face detection method based on dual-color spaces and an improved Xception," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 6, pp. 3527–3538, 2022.
- [14] G. Wang, Q. Jiang, and X. Jin, "FFR_FD: effective and Fast detection of DeepFakes based on feature point defects," 2021, <https://arxiv.org/abs/2107.02016>.
- [15] J. Yang, S. Xiao, A. Li, G. Lan, and H. Wang, "Detecting fake images by identifying potential texture difference," *Future Generation Computer Systems*, vol. 125, pp. 127–135, 2021.
- [16] L. Trinh, M. Tsang, and S. Rambhatla, "Interpretable and trustworthy deepfake detection via dynamic prototypes," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1973–1983, Waikoloa, Hawaii, USA, January 2021.
- [17] R. Caldelli, L. Galteri, I. Amerini, and A. Del Bimbo, "Optical Flow based CNN for detection of unlearned deepfake manipulations," *Pattern Recognition Letters*, vol. 146, pp. 31–37, 2021.
- [18] A. Rossler, D. Cozzolino, and L. Verdoliva, "Faceforensics++: learning to detect manipulated facial images," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1–11, Seoul, Korea (South), October 2019.
- [19] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," *European Conference on Computer Vision*, Springer, Berlin, Germany, 2006.
- [20] Y. Li, X. Yang, and P. Sun, "Celeb-df: a large-scale challenging dataset for deepfake forensics," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3207–3216, Seattle, WA, USA, June 2020.
- [21] B. Dolhansky, R. Howes, and B. Pflaum, "The deepfake detection challenge (dfdc) preview dataset," 2019, <https://arxiv.org/abs/1910.08854>.
- [22] E. Sabir, J. Cheng, and A. Jaiswal, "Recurrent convolutional strategies for face manipulation detection in videos," *Interfaces*, vol. 3, no. 1, pp. 80–87, 2019.
- [23] L. Li, J. Bao, and T. Zhang, "Face x-ray for more general face forgery detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5001–5010, Seattle, WA, USA, June 2020.
- [24] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1904–1916, 2015.
- [25] H. H. Nguyen, J. Yamagishi, and I. Echizen, "Use of a capsule network to detect fake images and videos," 2019, <https://arxiv.org/abs/1910.12467>.
- [26] S. Cao, Q. Zou, X. Mao, Y. Dengpan, and W. Zhongyuan, "Metric learning for anti-compression facial forgery detection," in *Proceedings of the 29th ACM International Conference on Multimedia*, pp. 1929–1937, Virtual Event, China, October 2021.
- [27] S. Chen, T. Yao, Y. Chen, S. Ding, J. Li, and R. Ji, "Local relation learning for face forgery detection," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 2, pp. 1081–1088, 2021.
- [28] H. Liu, X. Li, and W. Zhou, "Spatial-phase shallow learning: rethinking face forgery detection in frequency domain," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 772–781, Seattle, WA, USA, June 2021.
- [29] T. Zhao, X. Xu, and M. Xu, "Learning self-consistency for deepfake detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15023–15033, Montreal, Canada, October 2021.
- [30] Y. Zheng, T. Liu, Q. Li, and J. Li, "Integrated analysis of long non-coding RNAs (lncRNAs) and mRNA expression profiles

- identifies lncRNA PRKG1-AS1 playing important roles in skeletal muscle aging,” *Aging*, pp. 15044–15060, 2021.
- [31] A. Haliassos, K. Vougioukas, and S. Petridis, “Lips don’t lie: a generalisable and robust approach to face forgery detection,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5039–5049, Seattle, WA, USA, September 2021.
- [32] J. Li, H. Xie, and J. Li, “Frequency-aware discriminative feature learning supervised by single-center loss for face forgery detection,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6458–6467, Seattle, WA, USA, June 2021.
- [33] Z. Gu, Y. Chen, and T. Yao, “Spatiotemporal inconsistency learning for DeepFake video detection,” in *Proceedings of the 29th ACM International Conference on Multimedia*, pp. 3473–3481, Stockholm, Sweden, September 2021.
- [34] B. Zi, M. Chang, and J. Chen, “Wilddeepfake: a challenging real-world dataset for deepfake detection,” in *Proceedings of the 28th ACM international conference on multimedia*, pp. 2382–2390, Seoul, Korea, October 2020.

Research Article

Manipulated Faces Detection with Adaptive Filter

Chaoyang Peng , Tanfeng Sun , Zhongjie Mi, and Lihong Yao

School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, China

Correspondence should be addressed to Tanfeng Sun; tfsun@sjtu.edu.cn

Received 24 February 2022; Accepted 19 August 2022; Published 15 October 2022

Academic Editor: Jegatha Deborah

Copyright © 2022 Chaoyang Peng et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

With the progress of face manipulation techniques, synthesized faces are spreading on the Internet, which raises concerns about potential threats. To prevent these techniques' abuse, various detection algorithms have been proposed. In this paper, we consider the image's frequency information, then propose an adaptive filtering algorithm named spatial and adaptive filtering (SAF) Network. SAF is a dual-stream network that considers spatial and frequency domains. In the frequency domain, wavelet transform is used to divide the image into different frequency bands, then an adaptive filter is introduced, which aims to capture more decisive information by giving different weights to different frequencies. To fuse spatial and frequency features, spatial pyramid pooling fusion (SPPF) is proposed, which solves the mismatch of feature maps, and considers the relationship between different patches by attention mechanism. Experiment results show that the performance of SAF is better than the comparison algorithm.

1. Introduction

With the rapid development of Deepfakes technology, a large number of manipulated faces have emerged on the Internet. Similar to text semantics [1], images also have semantic information, so the content of images may be modified. From the forgery results, the tampering methods can be divided into two categories: tampering with some specific character attributes [2, 3] or generating an entire face [4].

In order to detect tampered faces, many forensics algorithms have been proposed. These algorithm can be roughly divided into three categories. First, detection based on biometrics [5, 6]. Second, detection based on spatial domain [7, 8]. Third, detection based on frequency domain [9–11]. Although the existing algorithm has achieved good detection results on public datasets, there are still some problems to be solved. On the one hand, new face manipulation methods are proposed constantly, and the quality of generated faces is higher and higher, which increases the difficulty of detection. Therefore, the detection ability of the previous algorithm may be reduced. On the other hand, the problem of detecting forged faces training and testing in the same dataset is already reasonably solved, so the real challenge is to train on one dataset but test on another with totally different methods.

The current detection algorithms basically focus on deep learning. Most of them use Convolutional neural network (CNN) [12, 13] to detect directly in the spatial domain. They regard deepfake detection as an image classification problem and use CNN to extract features. However, some image post-processing methods will reduce the performance [9, 14, 15] in the RGB domain, such as Gaussian noise, JPEG compression, and median filtering. In the frequency domain, previous work has used filters to preprocess the images. For example, Stuchi et al. [16] designed multiple frequency band filters to operate on the image and manually set the parameters based on experience. However, this method of manually designing filters is inappropriate in some situations because it is difficult for filters with fixed parameters to adaptively capture information of different frequencies.

This paper proposes an adaptive filter to solve the disadvantage of manual filters. Every color component of images is split into different frequencies, and then they are concentrated together to get a multi-channel input, so each channel represents a specific frequency band of a color component. Squeeze-and-Excitation Networks (SENet) [17] can assign weights to different channels to achieve an adaptive filter.

This paper designs a dual-stream network, with one branch used to extract spatial features and the other branch used to extract frequency features adaptively. In extracting the frequency features, we use wavelet transform. According to the properties of the wavelet transform, the image size will be reduced by half after the wavelet transform. So even if the same network is used for both branches, the size of the extracted features in the spatial and frequency domains will be different. If different networks are used, it is more challenging to ensure the consistency of the shape of the feature map.

Spatial pyramid pooling can solve the inconsistency problem of input images and get a fixed size output no matter how large the input image size is. In order to fuse the features extracted from the two branches, we propose spatial pyramid pooling fusion (SPPF). After SPPF, the spatial features and frequency features are fused, and finally, the fused features are passed through the fully connected layer to discriminate the results, real or fake.

2. Related Work

2.1. Manipulated Faces Generation. As for manipulated faces generation, there has been extensive research. Face2Face [18] is known as face reenactment, which modifies the facial expression of the target face. In Face2Face, an actor animates the facial expressions of the target video, and then a manipulated output video is generated. Neural Textures [19] also modifies the facial expression of the source actor. Feature maps are trained as part of the scene capture process, and the training process is end-to-end. StyleGAN [4] can learn high-level attributes automatically, which allows it finely control face properties. ICface [3] proposes a face animator, a data-driven system. It is implemented as a two-stage neural networks, which can mix information from multiple sources. Li et al. [20] introduced a deepfake-based method that solved some problems.

Manipulated faces datasets are the significant benchmark for detecting algorithms. Some popular datasets are listed here. FaceForensics++ (FF++) [21], Celeb-DF [20], Google DFD [22], DFFD [23], Deepforensics-1.0 (DF-1.0) [24] and DFDC [25]. Examples are shown in Figure 1.

2.2. Manipulated Faces Detection. In order to detect tampered faces, many forensics algorithms have been proposed. The simplest way is to start with biological features and look for defects in visual effects [5, 6, 26]. Li et al. [5] detected manipulated faces by blink. This method combines Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN), which can capture the feature of the human eye blinking in videos. Because eyes in training pictures are generally open, the blinking frequency of the actors in the obtained video will be lower than that of normal people. In addition, people's blinking mainly includes the eye-closing stage and eye-opening stage. These two stages are a gradual process, which is easily ignored in video generation. Li et al. [6] Detected the false face boundary. For the face change performed by deepfake, the edge area of the replacement face

will leave traces. Therefore, the authenticity of the face can be judged by detecting the area around the face. Haliassos et al. [26] used high-level semantic irregularities in mouth movement as a feature, which are common in manipulated videos.

The current detection algorithms basically focus on deep learning. Most of them use CNN to detect directly in the spatial domain. Li et al. [7] found a more noticeable difference between the real image and the manipulated image in the YCrCb domain compared with the RGB domain. In this method, the residuals of the YCrCb domain are used as input, and a classifier is trained by CNN. Liu et al. [8] proposed Gram-Net, which used the global texture features. It has strong robustness and generalization ability. CNN is good at classification tasks. For example, Wang et al. [15] directly used the Resnet50 pre-trained on the Imagenet as the backbone, achieving good detection performance. Gowda et al. [27] compared three neural net models and showed that the ensemble method works better.

Some algorithms also use frequency information for detection [9, 11, 16, 28]. Frank et al. [28] comprehensively investigated the characteristics of different GAN structures in the frequency domain. They found that noticeable grid artifacts will be introduced due to the upsampling. Qian et al. [9] extracted two kinds of frequency features, frequency aware decomposition (FAD) and local frequency statistics (LFS), then proposed the F³-Net. Stuchi et al. [16] designed multiple frequency band filters to operate on the image and manually set the parameters based on experience. However, this method of manually designing filters is inappropriate in some situations because it is difficult for filters with fixed parameters to adaptively capture information of different frequencies.

3. Proposed Method

3.1. Framework. The framework of the proposed algorithm is shown in Figure 2, which fuses spatial and frequency features. The whole process consists of four parts. (1) Pre-processing. The spatial domain image is wavelet transformed, and each color component is decomposed into approximation (LL), horizontal (LH), vertical (HL), and diagonal (HH), a total of 12 feature inputs. After the wavelet transform, the size of the image is halved. That is, if the input image size is $w \times h$, then the size of the wavelet image is $w/2 \times h/2$. (2) Feature extraction. The original spatial domain image is fed into the pre-trained Resnet50 network to extract the spatial features, and the wavelet transformed frequency domain image is fed into the pre-trained SE_Resnet to extract the frequency domain features. (3) Fusion. Since the size of the input frequency domain image is half of the spatial domain image, the size of the feature map obtained after feature extraction should also be halved. In order to make the feature map size consistent, spatial pyramid pooling (SPP) [29] is adapted. During the process of feature fusion, attention mechanism [30] is used here. (4) Classification. The fusion feature is flattened, then binary classification is performed by a fully connected layer. Finally, the detection result will be given.



FIGURE 1: Examples of manipulated faces (a) Celeb-DF[20] (b) FF++[21] (c) DFD[22] (d) DFFD[23] (e) DF-1.0[24] (f) DFDC[25].

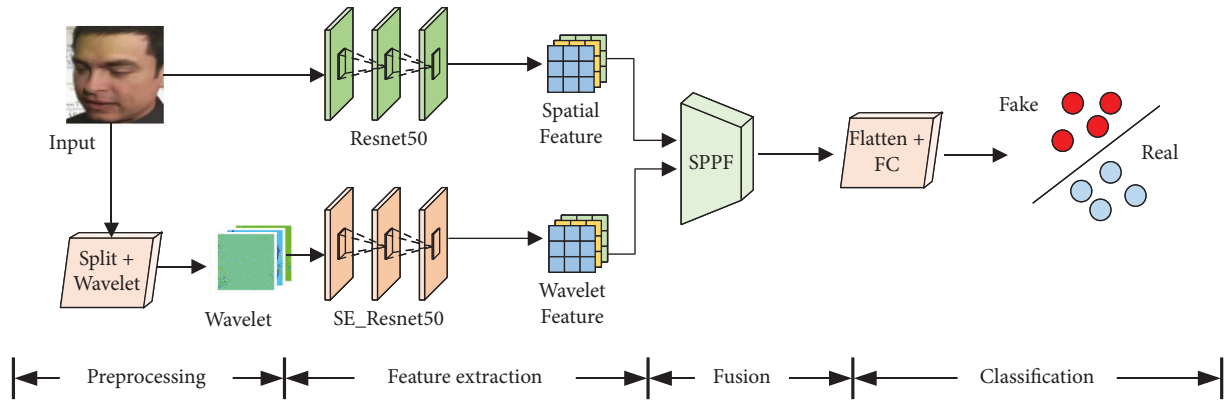


FIGURE 2: Framework of the proposed SAF. The input is a face to be detected. Spatial and frequency information is extracted by CNN, then they are fused by SPPF. The output is the result (real or fake).

3.2. Frequency Analysis. The natural image consists of three color components: R , G and B . But for the original image format inside the camera, each position has only one component. These colors are arranged in Bayer format [31]. Figure 3 shows the color matrix. Because the human eye is most sensitive to green light, the green component is the sum of the blue and red components. In order to convert the Bayer format to a natural image, CFA interpolation algorithm [31] is adopted. According to the principle of interpolation, the high-frequency information of the three components is similar. Given an image, wavelet transform is carried out. Figure 4 shows the scatter diagram of wavelet detail coefficients in HH. The three coordinate axes represent R , G , and B , respectively. The scatter diagram is distributed in a straight line, and the vector direction is $(1, 1, 1)$, which means that the high-frequency components are approximately equal.

For an image, each color component can be decomposed into high-frequency and low-frequency information (1)

$$C = C^l + C^h. \quad (1)$$

Because of the similarity of high-frequency components, the difference channel $C_1 - C_2$ can be represented by Equation (2). Therefore, for real images, the high-frequency component is filtered out. However, for manipulated faces, an interpolation algorithm is not adopted so that some high-frequency information will be left.

$$C_1 - C_2 = C_1^l + C_1^h - C_2^l - C_2^h \approx C_1^l - C_2^l. \quad (2)$$

3.3. Adaptive Filter. The image has low and high-frequency contents. Although manipulated faces already have sound visual effects, the details are still lacking, so the difference between real and manipulated faces is more evident in high-frequency. Based on this premise, we studied the imaging process of the camera and found that the high-frequency

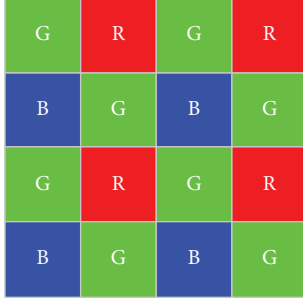


FIGURE 3: Bayer color filter array pattern.

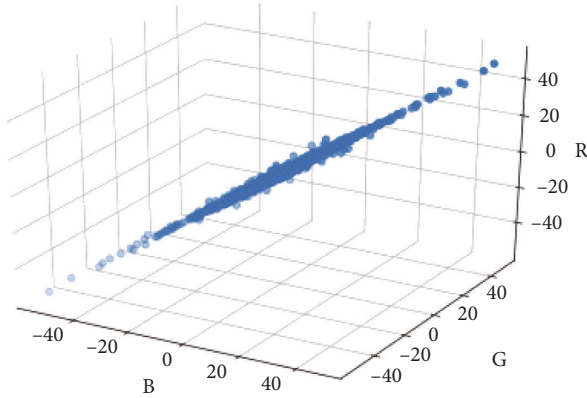


FIGURE 4: Scatter plots of detail wavelet coefficients.

components of different channels in natural images have a strong correlation, while this property is relatively weak in forged images. Therefore, the difference in high-frequency information is the key to our algorithm. Although the low-frequency information of the natural and forged images are relatively similar, it contains the central semantics of the images. For some manipulated faces with poor visual effects, they can be distinguished clearly with human eyes. So low-frequency information is also taken into account.

Section 1 shows that most previous works [16] design filters manually. This paper gives an adaptive filter. First, color components are split into R, G, and B. Then, high and low-frequency information from three channels is divided. Each color component is transformed into LL, LH, HL, and HH. LL is low-frequency information, while LH, HL, and HH are high-frequency information. To simplify the problem, the Haar wavelet is operated only once, so there will be a 12 channels image, and each channel represents a different frequency and color. SENet [17] considers the relationship between channels, which gives different weights to different channels. Therefore, an adaptive filter is achieved.

3.4. Spatial Pyramid Pooling Fusion. The process of spatial pyramid pooling fusion (SPPF) is shown in Figure 5. After it, the features of the two branches are fused. Even if the output sizes are inconsistent, the method can also realize the fusion. For Figure 5, several explanations are given here. (1) After feature extraction, there will be two feature maps. Here, it is

assumed that the dimensions are $M \times M \times \text{Ch1}$ and $N \times N \times \text{Ch2}$. Figure 5 shows that Ch1 is equal to 3 and Ch2 is equal to 2, which is only an example. (2) Spatial pyramid pooling [29] ignores the input size and compresses each channel into a vector whose length is L. For example, the length shown in Figure 5 is 4. (3) Using attention [30] to capture the global information and calculate the relationship between various regions. (4) two branches are mixed by multiplying each other, then stacked.

SPPF has two obvious advantages: (1) It solves the inconsistency of feature maps to realize fusion. (2) Since each element of SPP corresponds to a patch in the original map, the relationship between different patches can be reflected when using the attention mechanism, and the size of patches does not need to be the same.

Features of spatial (IS) and frequency (IF) are extracted by networks. For IS, Resnet50 is adopted here, shown in Equation (3). For IF, channels are divided firstly, then perform wavelet transform on them. Next, SE_Resnet50 is used to extract frequency information, shown in Equation (4) and (5).

$$IS_{M \times M \times \text{Ch1}} = \text{Resnet50}(\text{Input}_{R,G,B}), \quad (3)$$

$$IW = [\text{Wavelet}(\text{Input}_R), \text{Wavelet}(\text{Input}_G), \text{Wavelet}(\text{Input}_B)], \quad (4)$$

$$IS_{N \times N \times \text{Ch2}} = \text{SE_Resnet50}(IW). \quad (5)$$

Due to their different shapes, the extracted features need to be fused. When it comes to frequency features, the wavelet changes the size of the input image in half, so the size of the feature map of frequency is also half compared with spatial's. In addition, SPP levels (set to 4 in this paper) determine the times of pooling, and pooling type represents pooling mode (max-pooling is adopted). Here, the attention mechanism is used to capture the correlation between patches. After crossing the information of IS and IF, Fusion1 and Fusion2 have the same columns, so they can be concentrated to get the fusion feature (FF). The specific process is shown in Algorithm 1.

4. Experiment Analysis

4.1. Setup

4.1.1. Dataset. Manipulated image datasets are important benchmarks to evaluate the effect of the detection algorithm. In this paper, Celeb-DF [20], FaceForensics++ (FF++) [21] are selected.

- (i) Celeb-DF [20]: The second-generation Deepfakes dataset, containing 590 real and 5639 Deepfakes videos.
- (ii) FF++ [21]: FF++ is the most widely studied, which includes 1k real and 4k fake videos generated by four methods (Deepfakes (<https://github.com/deepfakes/faceswap>), Face2Face [18], FaceSwap

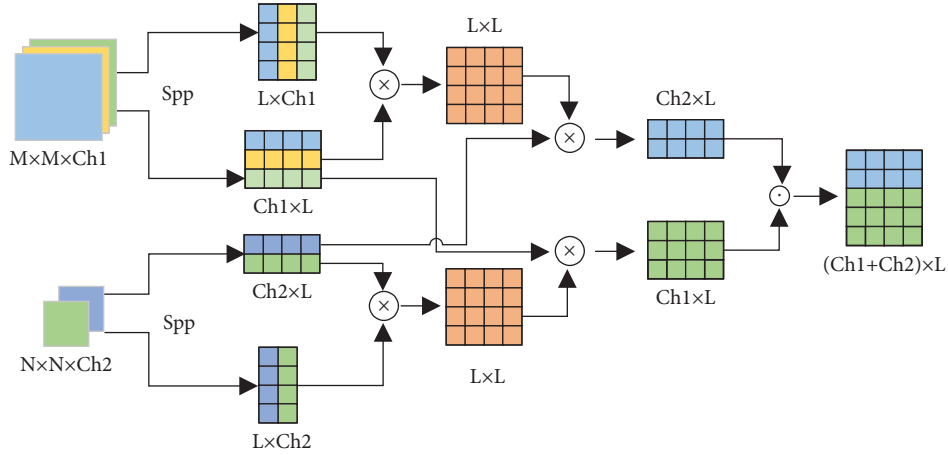


FIGURE 5: Process of spatial pyramid pooling fusion.

```

(i) Input: spatial feature  $IS$  ( $M \times M \times Ch1$ );
(ii) frequency feature  $IF$  ( $N \times N \times Ch2$ );
(iii) SPP levels  $L$ ;
(iv) pooling type  $T$ 
(v) Output: fusion feature  $FF$ 
(1)  $cnt = 0$ ;
(2)  $S = []$ ;
(3)  $F = []$ ;
(4) while  $cnt < L$  do
(5)    $cnt += 1$ ;
(6)    $Kernel\_S = (M/cnt, M/cnt)$ ;
(7)    $Kernel\_F = (N/cnt, N/cnt)$ ;
(8)    $S = [S, Pooling(IS, T, Kernel\_S)]$ ;
(9)    $F = [F, Pooling(IF, T, Kernel\_F)]$ ;
(10) end
(11)  $Attention\_S = S \cdot S^T$ ;
(12)  $Attention\_F = F \cdot F^T$ ;
(13)  $Fusion1 = Attention\_S \cdot F^T$ ;
(14)  $Fusion2 = Attention\_F \cdot S^T$ ;
(15)  $FF = [Fusion1, Fusion2]$ 

```

ALGORITHM 1: Spatial Pyramid Pooling Fusion.

(<https://github.com/MarekKowalski/FaceSwap/>) and Neural Textures [19]). All videos in FF++ have three resolutions: raw quality (c0), high-quality (c23), and low quality (c40).

- (iii) DFD: Google DFD [22] is the supplement of FF++, with 363 real and 3068 fake videos. They are generated by publicly available methods (<https://ai.googleblog.com/2019/09/contributing-data-to-deepfake-detection.html>)

4.1.2. Evaluation Metrics. The receiver operating characteristic curve can easily find out the recognition ability of a classifier to samples at a certain threshold. In this paper, The area under the curve (AUC) values are taken as an evaluation metric, which is widely used in manipulated face detection [9, 10, 32].

4.1.3. Implementation Details. Resnet50 is used as the spatial backbone in the experiments to extract features and is loaded with the imagenet pre-training model. For the adaptive frequency filter, to reuse Resnet50, the SE layer is added on top of Resnet50 to get SE_Resnet50, and the imagenet pre-training model is loaded. The learning rate is set as $5e-5$ of Adam optimizer. In this paper, the image input size is 224×224 , so for the spatial domain, the feature map shape is 7×7 , while the image size after wavelet transform is halved to 112×112 , so for the frequency branch, the feature map shape is 4×4 . Haar is selected as the wavelet basis, and the order is 1. The detailed output shape of every layer is listed in Table 1.

4.2. Intra-dataset Experiments. To prove the effectiveness of SAF, intra-dataset experiments are conducted. FF++ and Celeb-DF are selected as the test database.

4.2.1. Evaluation on FF++. FF++ is the most widely used dataset. Therefore, the proposed method is compared with the previous algorithm. Our experiments are conducted on high-quality (c23) videos, and all four types are used. Each methods provides 10k images, and the ratio of training set to testing set is 4:1. In the experiment, the positive and negative samples are balanced, that is, the ratio of real image to forged image is 1:1. Several recent works are compared with our method, including: i.e., (i) Face X-ray [33], which detects manipulated faces across blending boundaries, (ii) F³-Net [9], which uses frequency features as clues, (iii) Two Branch [34], which proposes a two-branch structure: original and frequency information, (iv) SPSL [10], which combines spatial image and phase spectrum to capture the upsampling artifacts, (v) EfficientNet-B4 (Eff-B4) [35], which is popular in the DeepFake Detection Challenge due to its performance, (vi) Capsule [36], which uses capsule network to detect spoofs, such as replay attacks and deep-fakes, (vii) Xception [21], which has good performance in manipulated faces detection and can significantly reduce the number of parameters, (viii) MaDD [32], which captures artifacts by multiple attentional map.

TABLE 1: Network structure.

Layer	Output size	
Input	224 × 224 × 3 (spatial)	112 × 112 × 12 (frequency)
Backbone	7 × 7 × 2048 (Resnet50)	4 × 4 × 2048 (SE_Resnet50)
SPPF	4096 × 30	
Flatten + FC	2	

TABLE 2: Evaluation on FF++.

Method	AUC score (%)
Face X-ray [37]	87.4
F ³ -net [9]	98.1
Two branch [34]	98.7
SPSL [10]	95.3
Eff-B4 [35]	99.2
Capsule [36]	96.6
Xception [21]	99.7
MaDD [32]	99.3
Ours	99.7

The results are shown in Table 2 and data are cited directly from [10, 32, 38]. The AUC of the proposed method achieves 99.4%, whose performance is better than the comparison algorithm. The AUC of Face X-ray is only 87.4%, and the proposed method is 12% higher than it. Xception [21] performs best in comparison methods, whose AUC is 99.7%. The proposed method can also reach it.

4.2.2. Evaluation on Celeb-DF. Compared with FF++, the forged videos in Celeb-DF have a better visual effect. So we conduct experiments on it. Due to the data imbalance, 60 and 8 are set as the sampling rates for real and manipulated faces respectively, which are set according to SE_EDNet [14]. Table 3 gives the comparison with previous methods. Capsule [36] has been introduced in the last section. I3D [33] is a spatiotemporal network whose convolution and pooling kernels are 3D. Triplet [39] uses a triplet network architecture to detect Deepfakes. SE_EDNet [14] use Euclidean distance to reflect the similarity between vectors, and a new calculation method of attention mechanism is proposed. EfficientNet-B4 (Eff-B4) [35] is popular in the DeepFake Detection Challenge due to its performance. Compared with these methods, the AUC of the proposed algorithm performs better, which achieves 99.9%.

4.3. Cross-Dataset Experiments. Although the proposed method outperforms the comparison algorithm, we have only made some slight improvements. As seen from section 3.2, the problem of detecting Deepfakes training and testing in the same dataset is already reasonably solved. The real challenge is to train on one dataset and test on another. The detection algorithm does not know the manipulated methods in the actual scene, so it is necessary to evaluate generalization. 16k images are sampled from Celeb-DF [20] (8k for real and 8k for forged), and the DFD [22] is same as it.

TABLE 3: Evaluation on Celeb-DF.

Method	AUC score (%)
Capsule [36]	93.2
I3D [33]	97.6
Triplet [39]	99.2
SE_EDNet [14]	99.7
Eff-B4 [35]	99.8
Ours	99.9

4.3.1. Cross-Dataset Evaluation on Celeb-DF. This section analyses the generalization ability of SAF on unseen data and gives the comparison results. The model is trained on FF++ (all four methods) but evaluated on Celeb-DF. The experimental results are shown in Table 4. Results of previous methods are directly cited from MaDD [32] or original papers. As demonstrated in Table 2, the proposed algorithm performs best in intra-dataset experiments compared to several published methods, whose AUC reaches 99.7%. Although the AUC score of Xception is equal to ours (shown in Table 2), it performs slightly worse than the proposed algorithm when testing on Celeb-DF. That is, the proposed algorithm has stronger transferability.

4.3.2. Cross-Dataset Evaluation on DFD. Besides Celeb-DF, we also conduct experiments on DFD [22]. The results are shown in Table 5, which are cited from [41]. FD² Net [41] use facial detail as the clue, which is the combination of light and identity texture. Table 5 indicates that the AUC of the proposed method reaches 84.8%, which outperforms previous algorithms. The previous algorithm with the strongest detection performance is FD² Net [41], but its AUC is still 5.7% lower than ours.

4.4. Ablation Study. Four sets of ablation experiments are conducted to analyze the effectiveness of wavelet adaptive filter and SPPF. Experiments results are shown in Table 6, and Δ refers to the difference in AUC score between Spatial.

- (i) Spatial. Spatial information (original image) is used as input and is sent to Resnet50 directly, which is the baseline.
- (ii) Wavelet. Wavelet image is used as input, which is sent to SE_Resnet50.
- (iii) Mixing + Cat. mixing wavelet and spatial information by cat, which simply combines the channels of dual-stream outputs.
- (iv) Mixing + SPPF. the input is same as Mixing + Cat, but SPPF is introduced to replace cat.

Three conclusions can be drawn from the results in Table 6: (1) proposed wavelet adaptive filter can detect manipulated faces well. When only using wavelet, although AUC (98.4%) is lower than Spatial (99.5%), it outperforms some previous methods in Table 2, such as Capsule [36] and I3D [33]. (2) Mixing spatial and wavelet information is helpful. It performs better than pure wavelet and pure spatial. (3) Proposed SPPF does better in fusing features than combining features directly by Cat.

TABLE 4: Cross-dataset evaluation on Celeb-DF.

Method	Celeb-DF
Face X-ray [37]	74.2
F ³ -net [9]	65.2
Two branch [34]	73.4
SPSL [10]	76.9
Eff-B4 [35]	64.3
Capsule [36]	57.5
Xception [21]	65.3
MaDD [32]	67.4
Ours	77.1

TABLE 5: Cross-dataset evaluation on DFD.

Method	Celeb-DF
Xception [21]	65.6
Eff-B4 ensemble [40]	72.8
FD ² net [41]	79.1
Ours	84.8

TABLE 6: Intra-dataset Experiments on Celeb-DF (*denotes baseline).

Method	AUC score (%)	Δ
Spatial*	99.5	—
Wavelet	98.4	-1.1
Mixing + Cat	99.7	0.2
Mixing + SPPF	99.9	0.4

5. Conclusion

This paper proposes a manipulated faces detection algorithm (SAF), which considers both spatial and frequency information. In the frequency domain, different frequencies are arranged into different channels, and then the channel weighting function of SENet is used for the adaptive filter. In addition, SPPF is proposed to fuse spatial and frequency features, which solves the problem of feature fusion of different shapes. Extensive experiments show the good detection and generalization ability of SAF.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

Ant Group of China supports the collaborative education project of industry university cooperation of the Ministry of Education (the second batch in 2021) (No. 21h010303219).

References

- [1] L. Jegatha Deborah, R. Baskaran, and A. Kannan, "Visualizing domain ontology using enhanced anaphora resolution algorithm," *International Journal of Database Management Systems*, vol. 3, no. 3, pp. 110–120, 2011.
- [2] H. Zhu, C. Fu, Q. Wu, W. Wu, C. Qian, and R. He, "Aot: appearance optimal transport based identity swapping for forgery detection," *Advances in Neural Information Processing Systems*, vol. 33, pp. 21699–21712, 2020.
- [3] S. Tripathy, J. Kannala, and E. Rahtu, "Icface: interpretable and controllable face reenactment using gans," in *Proceedings of the IEEE/CVF winter Conference on Applications of Computer Vision*, pp. 3385–3394, Snowmass, CO, USA, March 2020.
- [4] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8107–8116, Seattle, WA, USA, June 2020.
- [5] Y. Li, M.-C. Chang, and S. Lyu, "Ictu oculi: exposing ai created fake videos by detecting eye blinking," in *Proceedings of the 2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, pp. 1–7, IEEE, Hong Kong, China, December 2018.
- [6] Y. Li and S. Lyu, "Exposing deepfake videos by detecting face warping artifacts," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 46–52, Long Beach, CA, USA, June 2019.
- [7] H. Li, B. Li, S. Tan, and J. Huang, "Identification of deep network generated images using disparities in color components," *Signal Processing*, vol. 174, Article ID 107616, pp.1–12, 2020.
- [8] Z. Liu, X. Qi, and P. Torr, "Global texture enhancement for fake face detection in the wild," in *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8057–8066, June 2020.
- [9] Y. Qian, G. Yin, S. Lu, Z. Chen, and J. Shao, "Thinking in frequency: face forgery detection by mining frequency-aware clues," in *European Conference on Computer Vision*, pp. 86–103, Springer, Germany, 2020.
- [10] H. Liu, X. Li, W. Zhou, and Y. Chen, "Spatial-phase shallow learning: rethinking face forgery detection in frequency domain," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 772–781, Nashville, TN, USA, June 2021.
- [11] J. Li, H. Xie, J. Li, Z. Wang, and Y. Zhang, "Frequency-aware discriminative feature learning supervised by single-center loss for face forgery detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6458–6467, Nashville, TN, USA, June 2021.
- [12] C. Wang and W. Deng, "Representative forgery mining for fake face detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Article ID 14923, Nashville, TN, USA, June 2021.
- [13] F. Maher Salman and S. S. Abu-Naser, "Classification of real and fake human faces using deep learning," *International Journal of Applied Engineering Research*, vol. 6, no. 3, 2022.
- [14] C. Peng, L. Yao, T. Sun, X. Jiang, and Z. Mi, "Se_ednet: a robust manipulated faces detection algorithm," in *Computer Graphics International Conference*, pp. 80–88, Springer, Germany, 2021.
- [15] S.-Yu Wang, O. Wang, R. Zhang, A. Owens, and A. A. Efros, "Cnn-generated images are surprisingly easy to spot for now,"

- in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8695–8704, Nashville, TN, USA, June 2020.
- [16] J. A. Stuchi, M. A. Angeloni, R. F. Pereira et al., “Improving image classification with frequency domain layers for feature extraction,” in *Proceedings of the 2017 IEEE 27th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6, IEEE, Tokyo, Japan, September 2017.
- [17] J. Hu, Li Shen, and G. Sun, “Squeeze-and-excitation networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7132–7141, Nashville, TN, USA, June 2018.
- [18] J. Thies, M. Zollhöfer, M. Stamminger, C. Theobalt, and M. Nießner, “Face2face: real-time face capture and reenactment of rgb videos,” *Communications of the ACM*, vol. 62, no. 1, pp. 96–104, 2018.
- [19] J. Thies, M. Zollhöfer, and M. Nießner, “Deferred neural rendering: image synthesis using neural textures,” *ACM Transactions on Graphics*, vol. 38, no. 4, pp. 1–12, 2019.
- [20] Y. Li, X. Yang, Pu Sun, H. Qi, and S. Lyu, “Celeb-df: a large-scale challenging dataset for deepfake forensics,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3207–3216, Seattle, WA, USA, June 2020.
- [21] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Niessner, “Faceforensics++: learning to detect manipulated facial images,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1–11, Seoul, Korea (South), November 2019.
- [22] N. Dufour and A. Gully, “Contributing data to deepfake detection research,” *Google AI Blog*, vol. 1, no. 3, 2019.
- [23] H. Dang, F. Liu, Joel Stehouwer, X. Liu, and A. K. Jain, “On the detection of digital face manipulation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5781–5790, Seattle, WA, USA, June 2020.
- [24] L. Jiang, Li Ren, W. Wu, C. Qian, and C. C. Loy, “Deepforensics-1.0: a large-scale dataset for real-world face forgery detection,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2889–2898, Seattle, WA, USA, June 2020.
- [25] B. Dolhansky, J. Bitton, P. Ben et al., “The Deepfake Detection challenge (Dfdc) Dataset,” 2020, <https://arxiv.org/abs/2006.07397>.
- [26] A. Haliassos, K. Vougioukas, S. Petridis, and M. Pantic, “Lips don’t lie: a generalisable and robust approach to face forgery detection,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5039–5049, Nashville, TN, USA, June 2021.
- [27] A. Gowda and N. Thillaiarasu, “Investigation of comparison on modified cnn techniques to classify fake face in deepfake videos,” vol. 1, pp. 702–707, in *Proceedings of the 2022 8th International Conference on Advanced Computing and Communication Systems (ICACCS)*, vol. 1, IEEE, Coimbatore, India, March 2022.
- [28] J. Frank, T. Eisenhofer, L. Schönherr, and A. Fischer, “Leveraging frequency analysis for deep fake image recognition,” in *Proceedings of the 37th International Conference on Machine Learning*, pp. 3247–3258, PMLR, Germany, July 2020.
- [29] L. Shi, Z. Zhou, and Z. Guo, “Face anti-spoofing using spatial pyramid pooling,” in *Proceedings of the 2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 2126–2133, IEEE, Milan, Italy, January 2021.
- [30] X. Wang, R. Girshick, A. Gupta, and K. He, “Non-local neural networks,” in *Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7794–7803, Salt Lake City, UT, USA, June 2018.
- [31] B. Sun, N. Yuan, and Z. Zhao, “A hybrid demosaicking algorithm for area scan industrial camera based on fuzzy edge strength and residual interpolation,” *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 4038–4048, 2020.
- [32] H. Zhao, W. Zhou, D. Chen, T. Wei, W. Zhang, and N. Yu, “Multi-attentional deepfake detection,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2185–2194, Nashville, TN, USA, March 2021.
- [33] J. Carreira and A. Zisserman, “Quo vadis, action recognition? a new model and the kinetics dataset,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6299–6308, Honolulu, HI, USA, July 2017.
- [34] I. Masi, A. Killekar, R. M. Mascarenhas, S. P. Gurudatt, and W. AbdAlmageed, “Two-branch recurrent network for isolating deepfakes in videos,” in *European Conference on Computer Vision*, vol. 12352, pp. 667–684, Springer, Germany, 2020.
- [35] M. Tan and Q. Le, “Efficientnet: rethinking model scaling for convolutional neural networks,” in *Proceedings of the International Conference on Machine Learning*, pp. 6105–6114, PMLR, Long Beach, CA, USA, May 2019.
- [36] H. H. Nguyen, J. Yamagishi, and I. Echizen, “Capsule-forensics: using capsule networks to detect forged images and videos,” in *Proceedings of the ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2307–2311, IEEE, Brighton, UK, May 2019.
- [37] L. Li, J. Bao, T. Zhang et al., “Face x-ray for more general face forgery detection,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5001–5010, Seattle, WA, USA, June 2020.
- [38] J. Wang, Z. Wu, J. Chen, and Yu-G. Jiang, “M2tr: multi-modal multi-scale transformers for deepfake detection,” 2021, <https://arxiv.org/abs/2104.09770>.
- [39] A. Kumar, A. Bhavsar, and R. Verma, “Detecting deepfakes with metric learning,” in *Proceedings of the 2020 8th international workshop on biometrics and forensics (IWBF)*, pp. 1–6, IEEE, April 2020.
- [40] N. Bonettini, E. D. Cannas, S. Mandelli, L. Bondi, P. Bestagini, and S. Tubaro, “Video face manipulation detection through ensemble of cnns,” in *Proceedings of the 2020 25th International Conference on Pattern Recognition (ICPR)*, pp. 5012–5019, IEEE, Milan, Italy, January 2021.
- [41] X. Zhu, H. Wang, H. Fei, Z. Lei, and S. Z. Li, “Face forgery detection by 3d decomposition,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2929–2939, Nashville, TN, USA, June 2021.

Research Article

Multisemantic Path Neural Network for Deepfake Detection

Nan Wu , Xin Jin , Qian Jiang , Puming Wang , Ya Zhang , Shaowen Yao ,
and Wei Zhou 

Engineering Research Center of Cyberspace, Yunnan University, Kunming 650091, Yunnan, China

Correspondence should be addressed to Qian Jiang; jiangqian_1221@163.com

Received 14 May 2022; Revised 6 August 2022; Accepted 27 September 2022; Published 11 October 2022

Academic Editor: Beijing Chen

Copyright © 2022 Nan Wu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

With the continuous development of deep learning techniques, it is now easy for anyone to swap faces in videos. Researchers find that the abuse of these techniques threatens cyberspace security; thus, face forgery detection is a popular research topic. However, current detection methods do not fully use the semantic features of deepfake videos. Most previous work has only divided the semantic features, the importance of which may be unequal, by experimental experience. To solve this problem, we propose a new framework, which is the multisemantic pathway network (MSPNN) for fake face detection. This method comprehensively captures forged information from the dimensions of microscopic, mesoscopic, and macroscopic features. These three kinds of semantic information are given learnable weights. The artifacts of deepfake images are more difficult to observe in a compressed video. Therefore, preprocessing is proposed to detect low-quality deepfake videos, including multiscale detail enhancement and channel information screening based on the compression principle. Center loss and cross-entropy loss are combined to further reduce intraclass spacing. Experimental results show that MSPNN is superior to contrast methods, especially low-quality deepfake video detection.

1. Introduction

Automated video editing techniques have made great strides in the past few years with the development of deep learning. In particular, people have shown great interest in face manipulation. It is now easy to transfer facial expressions from one video to another based on generative adversarial networks (GANs) and autoencoders [1]. Even those who do not know deep learning can easily change one person's face to another in a few minutes [2], and a fake face is difficult for human eyes to distinguish. It is easy to change who the speaker is or what is said. While deepfake techniques bring benefits, there are hidden dangers.

These techniques open a new window for film and television. For example, dead movie stars can reappear through face manipulation, and people who do not exist in the real world can be created through GANs. Moreover, malicious attacks and revenge porn are a small part of malicious face manipulation. This also influences politics, such as by tampering with speech content and spreading fake news [3]. As a result, deepfake videos have attracted the interest of researchers, and methods to

detect whether a face has been manipulated have become paramount.

Deepfake videos can have at least three levels of forgery characteristics: microscopic, mesoscopic, and macroscopic. Microscopic features correspond to unseen differences, such as anomalies in small regions. Macroscopic or semantic features refer to the whole image semantics that the human eyes can feel. Mesoscopic features are seen in between. Afchar et al. [4] designed MesoNet to detect mesoscopic features. Current deepfake video detection methods do not take full advantage of these three levels of features. Usually, authenticity discrimination has been based only on high semantic features, and the performance needs improvement. It is possible to design a network that can integrate the three levels for deepfake detection. However, semantic segmentation methods that can ensure the improvement of accuracy have not yet been proposed. Similar work was based on the practical experience of feature hierarchy division. In addition, it is uncertain whether the weights of the three hierarchical features are the same.

Deepfake video detection methods have achieved accuracy of nearly 100% for high-quality videos, but their

accuracy for low-quality videos with high-compression rates needs to be improved [5]. For example, the accuracy of Xception [6] was 99.26% for the uncompressed FaceForensics++ dataset [7], but 72.93% at the C40 compression rate without pretraining. The high-compression rate makes the video very blurry and the forgery trace becomes unclear and not obvious, thereby becoming more difficult to distinguish the real video from the fake. Most videos on the Internet are compressed due to upload size limitations; thus, low-quality video forgery detection is significant. For this kind of video, we studied the commonly used H.264 video compression format, which includes inter and intraframe compression [8]. If only the original adjacent frames are removed through interframe compression, the accuracy will indeed be improved in theory. However, this will lead to inconsistencies with the creation requirements of benchmark datasets such as FaceForensics++, so we only use intraframe compression, which preserves the Y channel information on the YCrCb space and compresses the CbCr information as much as possible. Figure 1 shows the changes of different channels in an image at different compression rates. After comparative experimental analysis, we find that when the Y channel of the image is used as the input, the accuracy is higher than that when other channels are used. In addition, to highlight the high-frequency information of low-quality videos, multiscale detail enhancement was performed on images before channel separation. Based on the above two findings, we propose a deepfake detection method integrating different semantics in the network. We find no standard for semantic division from the aspect of channel level, but division from the aspect of the receptive field of the convolution kernel is reasonable.

When considering semantic level importance, instead of assigning weights manually, we use channel-spatial attention to assign them automatically. Therefore, a multichannel network with different receptive fields is proposed to integrate the features at different levels to capture forgery features. In constructing the neural network, the essential information is extracted through preprocessing and input to the network. We connect the feature maps of multiple pathways or semantics and automatically assign the weights to the three semantics through the channel-spatial attention module, perform feature fusion, and classify. We train and test our model on FaceForensics++ and DeepFake-TIMIT [9] and perform cross-validation on Celeb-DFv2 [10]. Experimental results show that our network has better accuracy than current methods, especially in low-quality deepfake video encoding.

This work makes the following contributions:

- (1) A multiscale detail enhancement method is introduced in deepfake detection. Fuzzy features are extracted from three Gaussian kernels, the residuals are calculated with the original image, and the detailed texture features of the forged image are highlighted;
- (2) Based on the study of video compression methods, the extraction of significant channel information assists in the detection of forged images with high-compression rates;
- (3) A multipath network for multisemantic information fusion is proposed. The three kinds of semantic information are automatically assigned weights by a channel-spatial attention module, and low, medium, and high semantic information of forged images can be effectively divided and interpreted;
- (4) Our method is evaluated on manipulated videos datasets. It performs well on the DeepFake-TIMIT and FaceForensics++ datasets and generalizes satisfactorily on Celeb-DFv2. The proposed preprocessing method can improve the detection of low-quality counterfeit videos, and the network can comprehensively capture different semantic information of images.

2. Related Work

We summarize current fake video generation methods, analyze deepfake detection methods, and introduce our method.

2.1. Deepfake Image Generation. Image generation techniques have developed rapidly over the past two decades, and methods such as StyleGAN [11] can produce fake images or videos that are credible to the eye. It is especially difficult to see traces of forgery after a video is compressed. Juefei-Xu et al. [12] produced a comprehensive report on counterfeiting generation and detection. Deep learning generation techniques of deepfake videos include autoencoders and GANs. Forgery methods can be categorized by the generated results as entire face synthesis, attribute manipulation, identity swap, and expression swap, as shown in Figure 2. Entire face synthesis generates a face that does not exist in the world. The input of these networks is a random vector, and the output is a realistic fake face image. Many models can be used, such as WGAN [13], StyleGAN, and PGGAN [14]. Attribute manipulation can modify the attributes of a person’s head, including simple attributes such as expression, hair color, and baldness and complex attributes such as gender, age, and the wearing of glasses. Classic examples are StarGAN [15] and STGAN [16]. Identity swapping, which replaces a face in a source image with a target’s face, has attracted much interest in recent years. Apps such as Zao [17] allow one to swap identities with a favorite star. Moreover, there are malicious attacks. Examples of identity swapping methods include FaceSwap [18] and CycleGAN [19]. Also known as face reconstruction, face-swapping is somewhat similar to identity swapping, replacing the source image’s facial expression with that of the target image’s facial expression, which include Face2Face [1] and A2V [20].

Methods of forgery generation include AAMS [21] for style transfer, SC-FEGAN [22] for image repair, and SAN [23] for super-resolution, but most of these methods are not the focus of face manipulation detection. According to the risk rank, identity swapping entails the most risk, followed by expression swap. Entire face synthesis and attribute manipulation are not very dangerous.

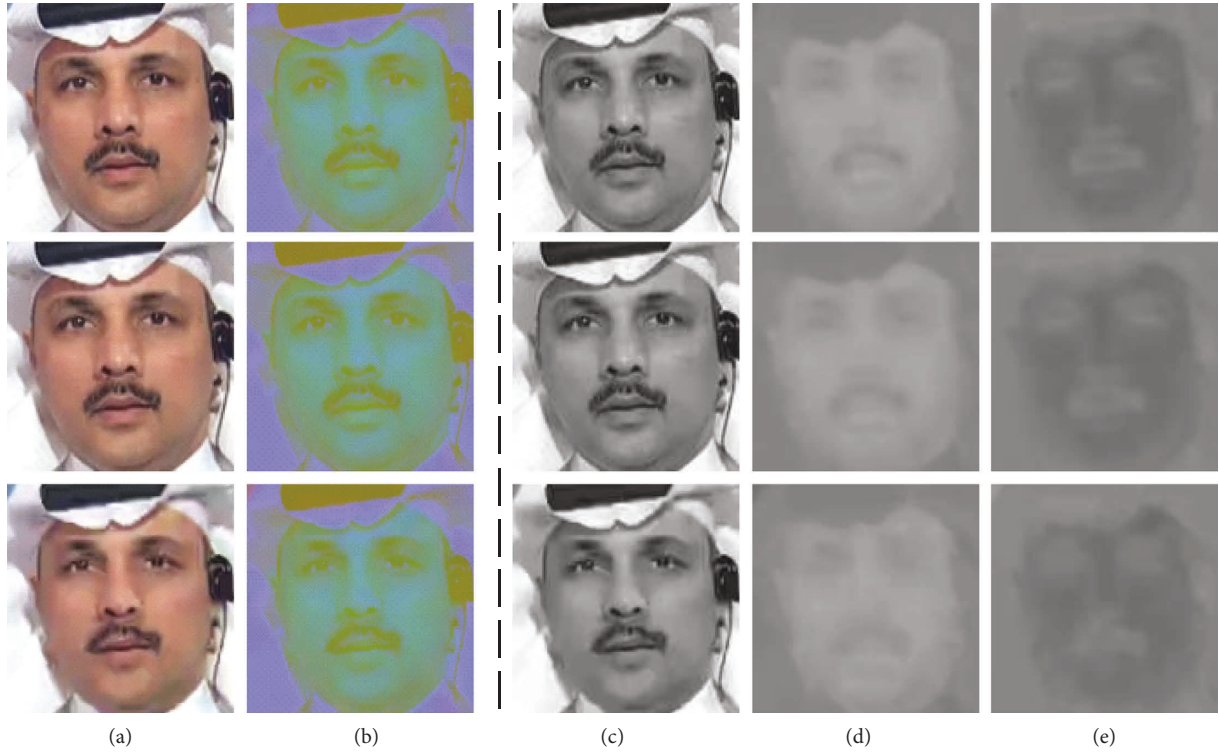


FIGURE 1: Changes in YCrCb channels of images with three compression ratios (declining video quality): (a) the image under the RGB channel; (b) the result of conversion to YCrCb channels; (c), (d), and (e) Y, Cr, and Cb images, respectively. Changes of Cr and Cb channel information are most apparent.



FIGURE 2: Samples of four forgery categories. Colors at lower-right indicate risk level.

2.2. *Deepfake Image Detection.* Methods to detect deepfake features are based on spatial or image pixels, the frequency domain, or biological signals. Spatial-based methods use either conventional feature forensics or deep learning.

Conventional image forensics relies on specific manipulation evidence [24], using frequency domain and statistical features such as local noise analysis, illumination, and device fingerprints to distinguish deepfakes. Nataraj et al. [25]

extracted co-occurrence matrices on three color channels in the pixel field and conducted classification training according to these features. Although the conventional forensics technique is mature, several shortcomings are present in dealing with deepfake videos because it pays more attention to abnormal features of local images. Deepfake videos are usually processed to avoid detection, such as by compression methods, compression rates, and condensation. Therefore, the conventional feature forensics technique cannot be directly applied to detect deepfake videos.

Methods based on or combined with deep learning have recently gained attention [26–29]. Sabir et al. [30] used recurrent neural networks to capture temporal differences in fake videos. Liu et al. [31] conducted an empirical study on real and fake faces and obtained some important findings. One of these findings is that the texture of a fake face is fundamentally different from that of a real face. Deep learning techniques and large datasets make it easier to catch the features associated with forgery [32]. This method can judge the authenticity of a single-frame image and detect video frames by a combined strategy, but it has limitations. Most learning models rely on the same dataset with the same data distribution for both training and testing and are weak in the face of unknown tampering types [33]. At the same time, the ability of deep learning models to detect highly compressed video frames is greatly reduced.

The method based on the frequency domain analyzes the differences of deepfake images such as through a Fourier or wavelet transform [34]. Durall et al. [35] proved that standard upsampling methods lead the forged images generated by these models to fail and to correctly reproduce the spectral distribution of natural training data. Most methods calculate feature maps with the differences between true and fake images in the frequency domain, and combine deep learning such as the support vector machine (SVM) for classification. Because the available spectrum of high-resolution images is much smaller than that of high-resolution photos, it is challenging to identify compressed videos.

Biometric authentication techniques have developed in recent years [36]. Detection methods based on biological signals cannot reproduce natural physiological characteristics by using fake videos, and the physiological characteristics of fake faces are inconsistent with those of real faces. [37]. Therefore, biological signal detection-based methods are constantly being developed by researchers. For example, by monitoring minimal periodic changes in skin color, Qi et al. [38] speculated that the normal heartbeat rhythm would be interrupted by deepfakes and proposed a dual temporal attention network. Although detection methods based on physiological signal characteristics can effectively make use of the defects of deepfake techniques, these methods gradually become invalid with the continuous improvement of generation methods, such as the addition of physiological characteristics (e.g., blink frequency). Besides, methods based on hard-to-find biological signals, such as heart rate, would be far less accurate due to video compression and other processing [39].

Because conventional forensic techniques are easily avoided by new deepfake techniques, frequency domain

feature-based statistical methods are not strong at detecting low-resolution forged videos, and biological signal-based methods are weak in improving generation technique. Most current work still adopts data-driven deep learning methods. As far as we know, current deep learning methods do not fully use the three semantics of images. For example, Mesonet only used mesoscopic semantics, while later networks used macroscopic semantics for judgment, such as Xception [7], FDFtnet [40], and AMTEN [41]. Zhao et al. [42] used microscopic and macroscopic semantics. Although some previous work mentioned semantics, they could not explain the relationship between network depth and the three types of semantics. Our work developed a targeted solution to this problem; specifically, the three semantics are set according to the width of the network, which has better interpretability. Moreover, ablation experiments show that the proposed method is effective and can surpass current methods at detecting forged images, especially in low-resolution videos. In addition, according to the compression principle, we propose a preprocessing method for low-resolution video.

3. Proposed Method

Based on the above analysis, we design a multisemantic path neural network (MSPNN) for deepfake detection to capture deepfake features under different semantics, as shown in Figure 3.

3.1. Multiscale Detail Enhancement. We use a multiscale approach to enhance the details of the source image. We first define three Gaussian filters:

$$G_1 = \begin{bmatrix} 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0219 & 0.0983 & 0.1621 & 0.0983 & 0.0219 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \end{bmatrix}, \quad (1)$$

$$G_2 = g_2 * g_2^T,$$

where $g_2 = [0.02760.06630.12380.18020.20420.18020.12380.06630.0276]$ and

$$G_3 = g_3 * g_3^T, \quad (2)$$

where $g_3 = [0.00810.01370.02200.03300.04650.06160.07660.09000.10150.09000.07660.06160.04650.06160.03300.02200.01370.0081]$.

Then, we obtain three fuzzy images using Gaussian image filters

$$\begin{aligned} B_1 &= G_1 \otimes I_{in}, \\ B_2 &= G_2 \otimes I_{in}, \\ B_3 &= G_3 \otimes I_{in}, \end{aligned} \quad (3)$$

where G_1 , G_2 , and G_3 are Gaussian kernels with respective kernel sizes of 5×5 , 9×9 , and 19×19 and standard

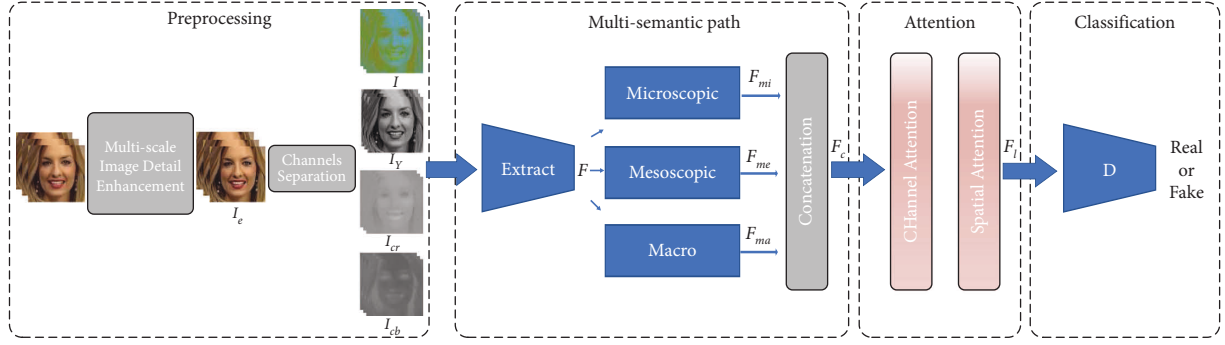


FIGURE 3: Cropped images of faces are used as input. After multiscale image detail enhancement preprocessing, I_e is obtained, and YCrCb channel separation is performed. The Y channel image I is taken as input. Preliminary feature F is extracted and put into the microscopic, mesoscopic, and macroscopic semantic channels. F_{mi} , F_{me} , and F_{ma} are obtained by feature extraction of the three channels. These are fused into the channel and spatial attention modules, and the weight of the three semantic feature maps is allocated. Results are input to the discriminator for classification.

deviations $\sigma_1 = 1.0$, $\sigma_2 = 2.0$, and $\sigma_3 = 4.0$; \otimes represents convolution; and B_1 , B_2 , and B_3 are the three filtered images. The fine, intermediate, and coarse details are, respectively, extracted as

$$\begin{aligned} D_1 &= I_{in} - B_1, \\ D_2 &= B_1 - B_2, \\ D_3 &= B_2 - B_3. \end{aligned} \quad (4)$$

We combine the three layers to generate a detailed image of the whole:

$$D^* = (1 - w_1 \times \text{sgn}(D_1)) \times D_1 + w_2 \times D_2 + w_3 \times D_3 + I_{in}. \quad (5)$$

According to experience, w_1 , w_2 , and w_3 are fixed as 0.5, 0.5, and 0.25, respectively. Figure 4 shows the process of image detail enhancement. Figure 5 shows the effect of multiscale detail enhancement. Faces at the top in Figure 5 are slightly blurred, while at the bottom, detail enhancement makes the visual perception of local details clearer, which aids in the detection of forged images with high compression.

3.2. Compressed Videos Analysis. According to our research, the detection accuracy of high- and medium-quality deep-fake videos, i.e., uncompressed and medium-compressed, respectively, is close to 100%, while that of high-compression videos is much worse, especially for some videos with more realistic tampering effects. Therefore, research on high-compression forged video must be improved. Since human eyes are not sensitive to the chromaticity of an image but are sensitive to its brightness, during image compression, it is desirable to retain as much chromaticity information as possible and compress brightness information to save storage space. Since the chrominance information of the compressed video hardly changes, the definition of the video does not change significantly. Since compression is carried out in YCrCb color space and our datasets are RGB images, spatial conversion is first required, given as follows:

$$\begin{bmatrix} Y \\ Cr \\ Cb \end{bmatrix} = \begin{bmatrix} 0.299 & 0.578 & 0.114 \\ 0.500 & -0.4187 & -0.0813 \\ -0.1687 & -0.3313 & 0.500 \end{bmatrix} \times \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix}, \quad (6)$$

where R , G , and B are the gray values of the three components of RGB.

Figure 1 shows images with different compression rates. The compression rate increases gradually from the first to the third row. The first line is the original image, and the image that is almost visually lossless in the second row is slightly compressed. The third row is a low-quality image. Column (a) shows images in RGB color space, and column (b) shows images under the YCrCb channels. Column c, column d, and column e show separate images using the YCrCb channels, such as the Y channel, the Cr channel, and the Cb channel, respectively. The change in the Y channel is the least obvious, and the change in the Cr and Cb channels is the most obvious. Inspired by the above observations, we extract the image information of the RGB channel into two types of luminance information and one type of chrominance information, i.e., the YCrCb channel. Then, we conducted four experiments using the Y channel, the Cr channel, the Cr channel, and the original image separately to verify our idea. Experimental results show that using only Y channel information can improve the accuracy of highly compressed video and has little effect on slightly compressed video.

3.3. Multisemantic Path. MSPNN can output feature maps with multiple semantics through different receptive fields and network depths. The features of these different layers are finally connected, and a learnable weight is added to the three feature layers for fusion classification. The final classification relies on the deep feature map and considers the shallow and middle feature maps. The overall framework is shown in Figure 3.

The network has three parts. First is simple image preprocessing to generate 32 feature maps. Different feature

maps are generated through three semantic channels. The network details are shown in Figure 6. Since low semantics can be understood as microscopic images, all filters in the semantic channel adopt a 3×3 window. The high semantics are the macroscopic features of the image, and the corresponding receptive field is more extensive, so the filter size of the semantic channel is 7×7 . Inspired by Inception [43], we replace a 7×7 convolutional kernel with three 3×3 convolutional kernels, which can reduce computation without reducing the receptive field and can have more nonlinear transformations, as shown in Figure 6. Mesoscopic semantics is between mesoscopic and macro semantics. The receptive field of this channel is 5×5 , and we use two 3×3 convolution kernels. Considering the influence of network depth semantics, the three semantic depths are also increased.

3.4. Semantic Integration. Although the microscopic, mesoscopic, and macroscopic semantics of images are juxtaposed, their importance is not the same. Hence, we apply a weight to each of the semantics instead of feeding back directly to the discriminator. In our model, these weights are learnable, which we accomplish through a channel-attention module to combine space and channels; this can achieve better results than SENet [44], which only pays attention to the channel. The first one is the channel-attention module of the image given as follows:

$$\begin{aligned} M_c(F) &= \sigma(\text{MLP}(\text{AvgPool}(F)) + \text{MLP}(\text{MaxPool}(F))) \\ &= \sigma(W_1(W_0(F_{\text{avg}}^c)) + W_1(W_0(F_{\text{max}}^c))), \end{aligned} \quad (7)$$

where σ denotes the sigmoid function, $W_0 \in \mathbb{R}^{C/r \times C}$ and $W_1 \in \mathbb{R}^{C/r \times C}$. Note that MLP weights W_0 and W_1 are shared for both inputs, and ReLU activation is followed by W_0 . Then the spatial attention is

$$\begin{aligned} M_s(F) &= \sigma(f^{7 \times 7}([\text{AvgPool}(F); \text{MaxPool}(F)])) \\ &= \sigma(f^{7 \times 7}([F_{\text{avg}}^s; F_{\text{max}}^s])), \end{aligned} \quad (8)$$

where $f^{7 \times 7}$ represents convolution with a 7×7 filter, and AvgPool() and MaxPool() are average and maximum pooling, respectively. The fused feature map is fed to the final classifier.

3.5. Loss Function. According to our investigation, the center loss function, while used in many face recognition tasks [45], does not improve performance in tasks such as handwritten number recognition. We conclude that the center loss function is more suitable for fine-grained classification tasks. To this end, we introduce a center loss function to our model as

$$L_c = \frac{1}{2} \sum_{i=1}^m \|x_i - c_{y_i}\|_2^2, \quad (9)$$

where $c_{y_i} \in \mathbb{R}^d$ represents the distribution center of y_i category data; that is, the feature center of true or fake faces,

x_i represents the feature before the full connection layer, and m is the batch size. We use this loss to continually decrease the sum of squares of the distance between the feature maps of each sample and the feature, i.e., to make the in-class distance as small as possible.

Normally, c_{y_i} should be updated as the depth features change. The choice of feature centers should consider the entire training set and average the features of each class in each iteration. Specifically, c_{y_i} is updated in small batches, and the centers are calculated by averaging the characteristics of the corresponding classes in each iteration. Second, to avoid large disturbances caused by a small number of mislabeled samples, we use the scalar α , which is limited to the range $[0, 1]$, to control the learning rate of the center. The updated equation of c_{y_i} is

$$\Delta c_j = \frac{\sum_{i=1}^m \delta(y_i = j) \cdot (c_j - x_i)}{1 + \sum_{i=1}^m \delta(y_i = j)}, \quad (10)$$

where if $y_i = j$ is satisfied, then $\delta(y_i = j) = 1$; otherwise, $\delta(y_i = j) = 0$; that is, when the tags y_i and C_j are of different categories j , then C_j does not require updating. We use a cross-entropy loss function and central loss joint supervision to train the network to learn true and fake features. The equation of the final loss function is given as follows:

$$\mathcal{L} = \mathcal{L}_s + \lambda \mathcal{L}_c. \quad (11)$$

We first consider L_s and L_c of equation (11) equally important, so we set λ as 1. Values can have different effects on the result, and we believe that multiple attempts can find a more suitable value. We compute

$$L_s = -\frac{1}{N} \sum_{i=1}^N [y_i \log(S(\hat{y}_i)) + \log(1 - y_i) \log(1 - S(\hat{y}_i))], \quad (12)$$

where \hat{y}_i is the score of the i -th face, and $y_i \in 0, 1$ is the related face label, where the label 0 is associated with faces from real, original videos, and 1 is associated with fake videos. N is the total number of faces used to train each batch, and $S(\cdot)$ is the sigmoid function.

4. Experimental Results and Analysis

We describe popular datasets, video segmentation methods, and their implementation, describe pretreatment ablation experiments and comparative experiments with other methods, and discuss verification of generalization.

4.1. Datasets. Our experiments use the FaceForensics++, DeepFake-TIMIT, and Celeb-DFv2 datasets. FaceForensics++ is one of the largest and most diverse deepfake datasets. It is a prominent face forgery dataset widely used in deepfake detection, with 1,000 YouTube videos. The authors of FaceForensics++ used four types of face tampering to create fake videos, including FaceSwap, DeepFakes, Face2-Face, and NeuralTextures. A total of 1000 deepfake videos are generated with each tampering method, including videos

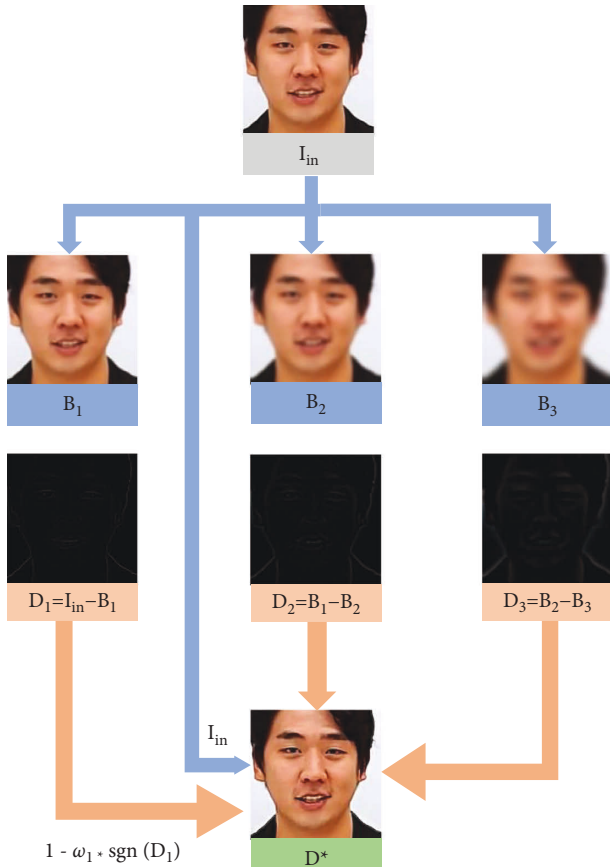


FIGURE 4: B_1 , B_2 , and B_3 are results of the three Gaussian filters. D_1 , D_2 , and D_3 are details of calculation of the original image and filtered results. The final image D^* is enhanced by incorporating details in the original image.

compressed with the original compression rate (C0), videos compressed with the micro compression rate (C23), and low-quality videos (C40). FaceForensics++ datasets have 1000 fake videos and 1000 real videos for each compression rate. When detecting forged videos, we divided the datasets into training, validation, and test sets according to the standard of FaceForensics++. There are 720 training sets, 140 validation sets, and 140 test sets.

DeepFake-TIMIT is generated by the face exchange algorithm based on the VidTIMIT dataset, which was developed using the faceswap-GAN method. Furthermore, Deepfake-TIMIT is the first deepfake dataset generated by GAN. The 640 generated fake videos are available in high (128×128) and low (64×64) quality. The production quality is better than that of Faceforensic++, but the video resolution is not high. We divided the dataset according to the settings of FaceForensics++. There are 320 videos of the two qualities, 230 training sets, 45 verification sets, and 45 test sets.

Celeb-DFv2 is a challenging deepfake video dataset that improves upon some weaknesses of other datasets. For example, UADFV, Faceforensic++, and Deepfake-TIMIT have low image resolution, poor quality of synthesized videos, rough tampering traces, and excessive flicker of video faces. The dataset consists of 590 real videos and 5,639

deepfake videos. Real videos from YouTube show celebrities of different genders, ages, and races.

For a fair comparison, we processed the video according to the clipping of FaceForensics++. All videos were framed, and dlib [46] was used to extract the feature points of each frame of the face to help locate and clip the face area, which was expanded by 1.3 times. Each video of the cropped face was taken in 30 frames. For data preparation of frame-level streams, we used OpenCV to extract frames. Since the datasets only operate on the faces in the video, not all frame information is helpful for deepfake detection from this perspective [7]. We focused our analysis on the area of the subject's face, and therefore on human faces, using dlib for face detection, which further reduced the amount of data processing. When extracting a face, dlib sometimes fails to recognize the face in a video frame, in whose case we skipped the frame and kept a constant number of faces captured in each video.

Figure 7 shows the input image samples and output feature maps in the three experiments. The first line uses the low-compressed DeepFakes datasets in FF++ for training and testing. The generation method of forged image in the second line is the same as in the first line, with a higher compression rate. The third line uses the DeepFakes datasets with low compression in FF++ for training and Celeb-DFv2 for testing so as to verify the generalization performance. The output feature maps are the result of the fusion of the three paths. It can be seen from Figure 7 that the real image with higher brightness is concentrated in the center of the feature map, while the forged image with higher brightness is concentrated in the lower part.

4.2. Implementation. All experiments were performed on RTX 3090. The baseline [7] has a high accuracy in uncompressed datasets, and we only evaluated our model on low- and high-compressed data. We implemented MSPNN using the PyTorch deep learning library. For more details, we selected cross-entropy as the loss function in the training phase. The output of the network was distributed between 0 and 1, and we adopted the autoadaptive algorithm Adam in the optimization process. The initial learning rate was $1e-4$, and the policies of cosine annealing LR were both used. The center loss function used the SGD optimizer. Batch normalization was used in each convolution to reduce the impact of overfitting. Dropout was introduced in the final full connection, with a ratio of 0.5. The batch size of the input data was 32. We trained our models with 100 epochs. The graph of the learning rate with each epoch was similar to a cosine function. The rest of the model settings were default values, the random seed was 43, and the input image was 224×224 .

4.3. Preprocessing Analysis. Preprocessing had two steps. Multiscale detail enhancement highlights face textures, especially low-quality images, which are so blurred that it is difficult to see forged traces. In this process, three filters of different sizes, G_1 , G_2 , and G_3 , were used to filter the image to obtain fuzzy images B_1 , B_2 , and B_3 . The original image was



FIGURE 5: Effect of multiscale detail enhancement. The top part shows the original images and the bottom part shows images with clear texture after multiscale detail enhancement.

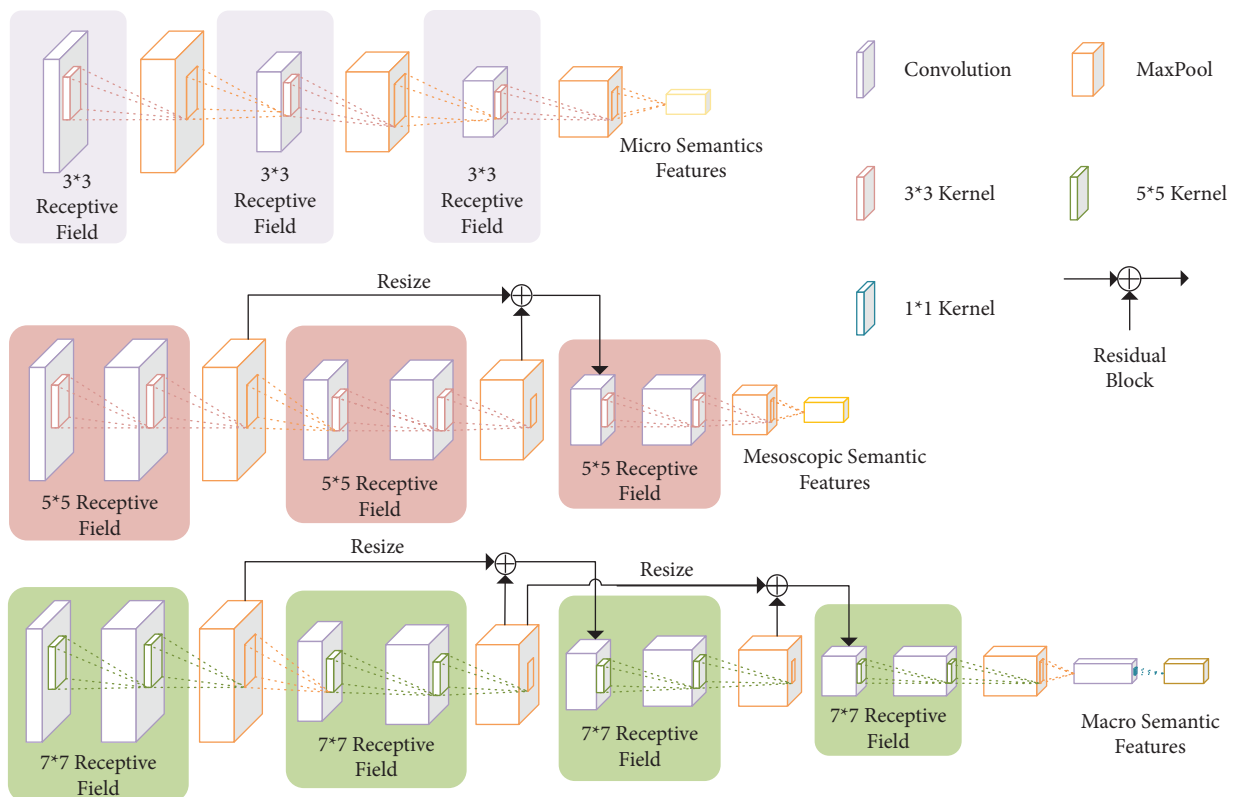


FIGURE 6: Network details of MSPNN. All receptive fields in micro semantic pathway are 3×3 . Superposition of two 3×3 convolution kernels replaces 5×5 receptive fields in the mesoscopic semantic path, and skip connection is used in the second block. Two 5×5 convolution kernels replace 7×7 receptive fields in the macroscopic semantic path. In the third and fourth blocks, skip connections reduce loss of information. Finally, the output of each path is aligned with other semantic feature maps through downsampling.

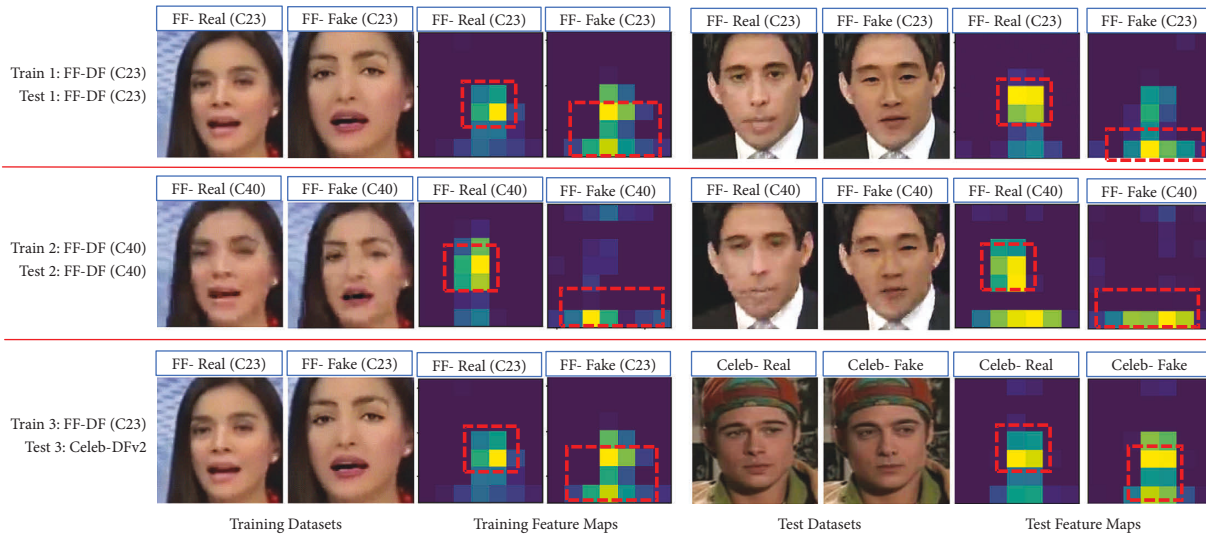


FIGURE 7: Training set, test set, and output feature maps. Red boxes indicate differences between real and fake images. It can be seen that the real image with higher brightness is concentrated in the center of the feature map, while the forged image with higher brightness is concentrated in the lower part of the feature map.

TABLE 1: Accuracy comparison of multiscale detail enhancement methods for datasets with different compression rates before and after introduction.

	Acc (%) on FF++(HQ)				Acc (%) on FF++(LQ)			
	DeepFakes	Face2Face	FaceSwap	NeuralTextures	DeepFakes	Face2Face	FaceSwap	NeuralTextures
Without detail enhancement	99.73	99.09	99.17	91.2	93.83	91.15	92.47	74.41
With detail enhancement	99.54	98.92	98.86	91.3	94.25	91.25	91.74	74.52

The bold values indicate the better results in the two experiments.

subtracted from B_1 to obtain detail image D_1 . The detail image D_2 was obtained by combining detail image D_1 and fuzzy image B_2 , and the detail image D_3 was obtained by combining detail image D_2 and fuzzy image B_3 . The three detail images were fused with the original image to enhance the detail images. The improved results are shown in Figure 5. Ablation experiments were performed on the datasets of FaceForensics++ with compression rates C23 and C40, as shown in Table 1, from which we can see that the detection performance of the high-compression dataset was effectively improved compared with the low-compression dataset, which shows the effectiveness of the proposed preprocessing method for low-quality datasets. It is worth noting that the proposed detection was improved at any compression rate on the most challenging NeuralTextures dataset. The proposed method only modifies the facial expression corresponding to the mouth, leaving the eye area unchanged, and requiring more subtle detection methods.

The second preprocessing step was channel separation for high-compression images with low detection accuracy. We investigated the video compression standard H.264 and found that the measure keeps the information of the Y channel as much as possible while compressing the other two channels. In Figure 1, we can see the changes in the knowledge of the three channels after compression. So we converted the RGB image to a YCrCb image, and the images of Y, Cr, and Cb channels were taken out for training. We found that the accuracy of the image

containing the brightness information channel is much higher than that containing the chroma information channel. The accuracy of the chromaticity information channel is much lower than that of the ordinary RGB channel, as shown in Table 2, according to which most subset accuracy can be improved by using only Y channel information on the FaceForensics++ dataset, especially on the highly compressed C40 dataset. The experimental effect on some datasets becomes worse, but this change is not very large. We believe that the forged image with a low compression rate is close to the original image, so the effect is not apparent.

4.4. Experimental Results. Most detection methods are based on macroscopic semantics, i.e., the final feature maps of the network. The difference between a natural face and a fake is often subtle and occurs in the local area. Minor artifacts caused by the deepfake method are usually stored in the shallow characteristic of texture information. We believe that the microscopic semantic or superficial semantic features cannot be ignored. Focusing only on details is also flawed. A microscopic analysis based on image noise cannot be applied to the compressed video environment, where the image noise is strongly reduced. It is difficult for the human eye to distinguish the forged images at the same higher semantic level, especially in fine-grained analyzes, such as face discrimination. Therefore, our work takes into account the three kinds of semantic information, which receptive

TABLE 2: Y, Cr, Cb, and RGB channels used as input for training and test results for C23 and C40 datasets.

	Acc (%) on FF++(HQ)				Acc (%) on FF++(LQ)			
	DeepFakes	Face2Face	FaceSwap	NeuralTextures	DeepFakes	Face2Face	FaceSwap	NeuralTextures
RGB	99.73	99.09	99.17	91.2	93.83	91.15	92.47	74.41
Cr channel	84.46	85.58	79.04	83.35	79.67	75.73	75.32	63.32
Cb channel	87.19	85.17	77.08	80.11	80.45	72.91	69.65	59.83
Y channel	99.57	99.21	99.34	91.08	94.35	91.01	92.55	74.92

Bold values indicate the best results for four different channels.

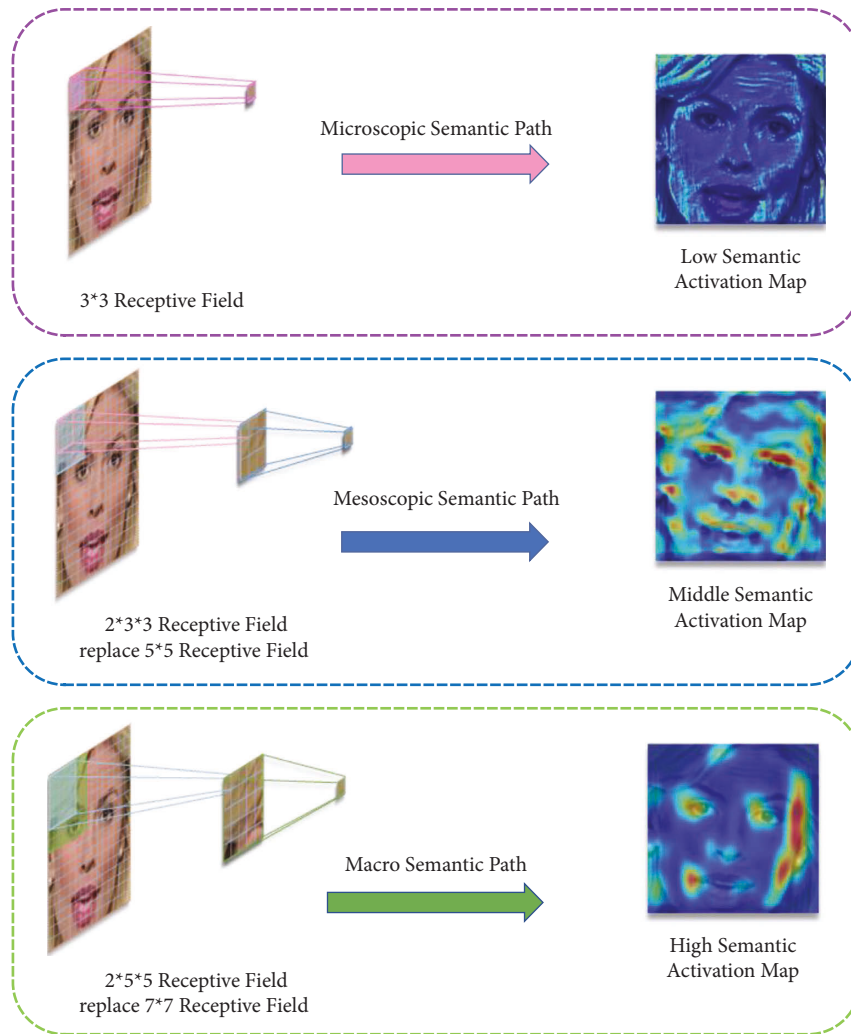


FIGURE 8: Receptive field description maps of different semantic paths and generated results.

fields of various sizes can also capture, and which are more explanatory, as shown in Figure 8.

Gradient-weighted class activation mapping is used to visually display the details of the attention of the three pathways, and it is evident that the microscopic semantic pathway pays more attention to details and the mesoscopic semantic path to multiple blocks. Macro semantics focus more on areas that are difficult to forge, such as the eyes, nose, and mouth because these are the most difficult to reproduce during the generation of forged images. In addition, small convolution kernels are used to map large convolution kernels, which reduces the computation of

convolution, while increasing multiple nonlinear activations, and the receptive field is unchanged. We add residual blocks to the mesoscopic and macroscopic semantic pathways to ensure that information is not lost when the network depth increases. In Table 3, we can observe that much of the accuracy of the datasets is improved under the three pathways. They have a poor effect on some datasets, in particular the NeuralTextures dataset, which only tampers with parts of the images, whereas our microscopic semantic pathway captures much information that is not helpful to the detection of these datasets. Our addition of preprocessing makes up for this problem, as shown in Table 1. We also

TABLE 3: Comprehensive precision comparison of three semantic pathways on FF++ dataset, and comprehensive precision comparison of three semantic pathways and multiple semantic pathway networks on neuraltextures, deepfakes, FACE2FACE, faceswap, and neuraltextures.

	Acc (%) on FF++ (HQ)				Acc (%) on FF++ (LQ)			
	DeepFakes	Face2Face	FaceSwap	NeuralTextures	DeepFakes	Face2Face	FaceSwap	NeuralTextures
Microscopic path	99.05	98.57	98.39	90.05	91.26	89.72	90.01	73.4
Mesoscopic path	99.2	99.17	98.86	91.01	92.32	88.92	88.20	73.52
Macroscopic path	99.32	99.21	98.77	91.52	94.4	91.04	92.41	75.09
Multipath	99.73	99.24	99.17	91.30	93.83	91.15	92.47	74.41

Results in bold indicate the best results of the four ablation experiments.

TABLE 4: Ablation experiment on assigning weight to different semantics by adding attention module (ACC %).

	Acc (%) on FF++(HQ)				Acc (%) on FF++(LQ)			
	DeepFakes	Face2Face	FaceSwap	NeuralTextures	DeepFakes	Face2Face	FaceSwap	NeuralTextures
Without attention	99.73	99.09	99.17	91.2	93.83	91.15	92.47	74.41
With attention	99.49	99.35	99.06	91.31	94.87	91.26	91.13	74.75

Bold values indicate higher results in two experiments.

TABLE 5: Quantitative detection results of ACC (%) using FF++ dataset on high quality (C23 light compression) and low quality (C40 heavy compression) videos and AUC on TIMIT datasets. Bold font indicates the best result.

	Acc on FF++(HQ)	Acc on FF++(LQ)	AUC on TIMIT(HQ)	AUC on TIMIT(LQ)
Bayar and stamm [47]	88.68	61.6	86.50	88.33
InMesonet [4]	57.81	69.75	81.15	82.63
Rahmouni et al. [48]	—	58.10	—	—
Mesonet [4]	54.91	50.28	63.68	77.44
Zhou et al. [49]	—	—	73.5	83.5
Chollet [6]	91.87	72.93	93.64	88.24
Nirkin et al. [50]	—	75.00	—	—
Ours	94.21	76.31	99.12	99.52

TABLE 6: Face cross test results, using frame-level AUC (%) to compare our method with others on both benchmarks.

Method	FF-DF	Celeb-DFv2
Two-stream [49]	70.1	53.8
Meso4 [4]	84.7	54.8
MesoInception4 [4]	83.0	53.6
HeadPose [51]	47.3	54.6
FWA [52]	80.1	56.9
DSP-FWA [52]	93.0	64.0
VA-MLP [53]	66.4	55.0
VA-LogReg [53]	78.0	55.1
Xception [6]	93.65	64.52
Multitask [54]	76.3	54.3
Two branch [55]	93.18	73.41
Capsule [56]	96.6	57.5
Ours	96.7	66.7

Bold values indicate higher results in two experiments.

conducted experiments to verify the effectiveness of our proposed channel and spatial attention modules. It is valid for most datasets, as shown in Table 4. In particular, we find that NeuralTextures and Face2Face can have satisfactory effects in the most complex datasets of FaceForensics++.

Our overall accuracy on FaceForensics++ datasets exceeds that of many other previous methods, as shown in Table 5. Most of the work on the TIMIT dataset uses the

AUC indicator. To evaluate the overall detection performance, we calculated the area under the curve (AUC), which is the area under the receiver operating characteristic (ROC) curve, whose maximum value is 1 and displays the results in Table 5. The AUC of our proposed method is higher than that of other methods, indicating better performance on compressed deepfake video detection.

4.5. *Validation of Generalization on Celeb-DFv2.* Cross-dataset validation was carried out to evaluate the generalization ability of the proposed method. The model was trained on FaceForensics++ and tested on Celeb-DFv2. We followed the setup of Celeb-DFv2 [10] to divide the test set and displayed the experimental index AUC scores in Table 6. It can be seen from the results that this method has a better generalization effect than most methods. Masi’s [55] generalization on Celeb-DFv2 is better than ours, but the AUC score in the original dataset is far behind. Our approach has limitations, but it has always been a challenge to balance accuracy and generalization.

5. Conclusion

Although methods for deepfake detection of videos and images have made much progress, few methods consider multiple aspects of semantic information. This work

proposes a new face forgery detection method, MSPNN, which can simultaneously capture micro, mesoscopic, and macro semantics to comprehensively distinguish forged images, with weights assigned automatically to the three semantics. The neural network can comprehensively capture different semantic information of an image. In view of the challenges of face tampering in a small-range, high-compression dataset, and cross-dataset, the proposed framework can effectively capture minor forged artifacts and macro forged traces, which can further improve the detection of high-compression forged images. This framework has good generalization as well. Furthermore, the proposed pre-processing method can improve the detection ability of our framework for low-quality counterfeit videos. Our future work will consider the combination of frequency domain information and brightness information at the separation point to integrate the corresponding features for deepfake detection.

Data Availability

The data supporting this work are from previously reported studies and datasets, which have been cited. The processed data are available at <https://github.com/ondyari/FaceForensics/blob/master/dataset/README.md>, <https://conradsanderson.id.au/vidtimit/#downloads> and <https://github.com/yuezunli/celeb-deepfakeforensics>.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this article.

Acknowledgments

This study was supported by the National Natural Science Foundation of China (Nos. 62002313, 62101481, 62166047), Key Areas Research Program of Yunnan Province in China (No. 202001BB050076), Major Scientific and Technological Project of Yunnan Province (No. 202202AD080002), Yunnan Fundamental Research Projects of China (Nos. 202201AU070033, 202201AT070112), the fund project of Yunnan Province Education Department (No. 2022j0008), Key Laboratory in Software Engineering of Yunnan Province (No. 2020SE408), the open project of Engineering Research Center of Cyberspace in 2021-2022 (No. KJAJQ202112012), and Industrial Internet Security Situation Awareness Platform of Yunnan Province.

References

- [1] B. Chen, T. Li, and W. Ding, "Detecting deepfake videos based on spatiotemporal attention and convolutional lstm," *Information Sciences*, vol. 601, 2022.
- [2] H. Li, W. Wang, C. Yu, and S. Zhang, "Swapinpaint: identity-specific face inpainting with identity swapping," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 7, 2021.
- [3] Y. Luo, F. Ye, B. Weng, S. Du, and T. Huang, "A novel defensive strategy for facial manipulation detection combining bilateral filtering and joint adversarial training," *Security and Communication Networks*, vol. 2021, no. 7, Article ID 4280328, 10 pages, 2021.
- [4] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, "Mesonet: a compact facial video forgery detection network," in *Proceedings of the 2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, pp. 1–7, Hong Kong, China, December 2018.
- [5] B. Han, X. Han, H. Zhang, J. Li, and X. Cao, "Fighting fake news: two stream network for deepfake detection via learnable srm," *IEEE Transactions on Biometrics, Behavior, and Identity Science*, vol. 3, no. 3, pp. 320–331, 2021.
- [6] F. Chollet, "Xception: deep learning with depthwise separable convolutions," in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1800–1807, Honolulu, HI, USA, July 2017.
- [7] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Niessner, "Faceforensics++: learning to detect manipulated facial images," in *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1–11, Seoul, Korea, November 2019.
- [8] T. Huang, X. Zhang, W. Huang, L. Lin, and W. Su, "A multi-channel approach through fusion of audio for detecting video inter-frame forgery," *Computers & Security*, vol. 77, pp. 412–426, 2018, [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0167404818304243>.
- [9] P. Korshunov and S. Marcel, "Deepfakes: a new threat to face recognition? assessment and detection," *CoRR*, vol. abs/1812.08685, 2018, [Online]. Available: <http://arxiv.org/abs/1812.08685>.
- [10] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, "Celeb-df: a large-scale challenging dataset for deepfake forensics," in *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3204–3213, Seattle, WA, USA, June 2020.
- [11] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4396–4405, Long Beach, CA, USA, June 2019.
- [12] F. Juefei-Xu, R. Wang, Y. Huang, Q. Guo, L. Ma, and Y. Liu, "Countering malicious deepfakes: survey, battleground, and horizon," *CoRR*, vol. abs/2103.00218, 2021, [Online]. Available: <https://arxiv.org/abs/2103.00218>.
- [13] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," 2017.
- [14] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," 2018.
- [15] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, "Stargan: unified generative adversarial networks for multi-domain image-to-image translation," in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8789–8797, Salt Lake City, UT, USA, June 2018.
- [16] M. Liu, Y. Ding, M. Xia et al., "Stgan: a unified selective transfer network for arbitrary image attribute editing," in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3668–3677, Long Beach, CA, USA, June 2019.
- [17] "Zao app," 2019, [Online]. Available: <https://zao-app.com>.
- [18] "Faceswap," 2016, [Online]. Available: <https://github.com/deepfakes/faceswap>.

- [19] J. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” *CoRR*, vol. abs/1703.10593, 2017, [Online]. Available: <http://arxiv.org/abs/1703.10593>.
- [20] S. Suwajanakorn, S. M. Seitz, and I. Kemelmacher-Shlizerman, “Synthesizing obama: learning lip sync from audio,” *ACM Transactions on Graphics*, vol. 36, no. 4, pp. 1–13, 2017.
- [21] Y. Yao, J. Ren, X. Xie, W. Liu, Y.-J. Liu, and J. Wang, “Attention-aware multi-stroke style transfer,” in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1467–1475, Long Beach, CA, USA, June 2019.
- [22] Y. Jo, J. Park, and Sc-fegan, “Face editing generative adversarial network with user’s sketch and color,” in *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1745–1753, Seoul, Korea, November 2019.
- [23] T. Dai, J. Cai, Y. Zhang, S.-T. Xia, and L. Zhang, “Second-order attention network for single image super-resolution,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, June 2019.
- [24] B. Chen, W. Tan, G. Coatrieux, Y. Zheng, and Y.-Q. Shi, “A serial image copy-move forgery localization scheme with source/target distinguishment,” *IEEE Transactions on Multimedia*, vol. 23, pp. 3506–3517, 2021.
- [25] L. Nataraj, T. M. Mohammed, S. Chandrasekaran et al., “Detecting gan generated fake images using co-occurrence matrices,” *Electronic Imaging*, vol. 2019, no. 5, pp. 532–537, 2019.
- [26] J. Yang, A. Li, S. Xiao, W. Lu, X. Gao, and Mtd-net, “MTD-Net: learning to detect deepfakes images by multi-scale texture difference,” *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 4234–4245, 2021.
- [27] J. Yang, S. Xiao, A. Li, W. Lu, X. Gao, and Y. Li, “Msta-net: forgery detection by generating manipulation trace based on multi-scale self-texture attention,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 7, pp. 4854–4866, 2022.
- [28] H. Ling, J. Huang, C. Zhao, Y. Yao, J. Chen, and P. Li, “Learning diverse local patterns for deepfake detection with image-level supervision,” in *Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7, Shenzhen, China, July 2021.
- [29] J. Hu, S. Wang, and X. Li, “Improving the generalization ability of deepfake detection via disentangled representation learning,” in *Proceedings of the 2021 IEEE International Conference on Image Processing (ICIP)*, pp. 3577–3581, Anchorage, AK, USA, September 2021.
- [30] E. Sabir, J. Cheng, A. Jaiswal, W. AbdAlmageed, I. Masi, and P. Natarajan, “Recurrent convolutional strategies for face manipulation detection in videos,” 2019.
- [31] Z. Liu, X. Qi, and P. H. Torr, “Global texture enhancement for fake face detection in the wild,” in *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8057–8066, Seattle, WA, USA, June 2020.
- [32] P. Chen, J. Liu, T. Liang et al., “Dlfnnet: end-to-end detection and localization of face manipulation using multi-domain features,” in *Proceedings of the 2021 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6, Shenzhen, China, July 2021.
- [33] I. Huseynli and S. Varli, “Analyzing deep learning models’ generalization ability under different augmentations on deepfake datasets,” in *Proceedings of the 2021 6th International Conference on Computer Science and Engineering (UBMK)*, pp. 694–698, Ankara, Turkey, September 2021.
- [34] G. Jia, M. Zheng, C. Hu et al., “Inconsistency-aware wavelet dual-branch network for face forgery detection,” *IEEE Transactions on Biometrics, Behavior, and Identity Science*, vol. 3, no. 3, pp. 308–319, 2021.
- [35] R. Durall, M. Keuper, and J. Keuper, “Watch your up-convolution: cnn based generative deep neural networks are failing to reproduce spectral distributions,” in *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7887–7896, Seattle, WA, USA, June 2020.
- [36] L. Li, C. Chen, L. Pan, J. Zhang, and Y. Xiang, “Sok: an overview of ppg’s application in authentication,” 2022, [Online]. Available: <https://arxiv.org/abs/2201.11291>.
- [37] S. Makowski, P. Prasse, D. R. Reich, D. Krakowczyk, L. A. Jäger, and T. Scheffer, “Deepyedentificationlive: oculomotoric biometric identification and presentation-attack detection using deep neural networks,” *IEEE Transactions on Biometrics, Behavior, and Identity Science*, vol. 3, no. 4, pp. 506–518, 2021.
- [38] H. Qi, Q. Guo, F. Juefei-Xu et al., “Deeprrhythm: exposing deepfakes with attentional visual heartbeat rhythms,” in *Proceedings of the 28th ACM International Conference on Multimedia, ser. MM ’20*, pp. 4318–4327, Association for Computing Machinery, New York, NY, USA, 2020.
- [39] X. Jin, D. Ye, and C. Chen, “Counteracting spoof: towards detecting deepfake with multidimensional biological signals,” *Security and Communication Networks*, vol. 2021, no. 1, , Article ID 6626974, 8 pages, 2021.
- [40] H. Jeon, Y. Bang, and S. S. Woo, “Fdfnet: facing off fake images using fake detection fine-tuning network,” in *ICT Systems Security and Privacy Protection*, M. Hölbl, K. Rannenberg, and T. Welzer, Eds., pp. 416–430, Springer International Publishing, Cham, 2020.
- [41] Z. Guo, G. Yang, J. Chen, and X. Sun, “Fake face detection via adaptive manipulation traces extraction network,” *Computer Vision and Image Understanding*, vol. 204, 2021, [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S107731422100014X>, Article ID 103170.
- [42] H. Zhao, W. Zhou, D. Chen, T. Wei, W. Zhang, and N. Yu, “Multi-attentional deepfake detection,” in *Proceedings of the IEEE/CVF. Conference on Computer Vision and Pattern Recognition*, Nashville, TN, USA, December 2021.
- [43] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2818–2826, Las Vegas, NV, USA, June 2016.
- [44] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, “Squeeze-and-excitation networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 8, pp. 2011–2023, 2020.
- [45] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, “A discriminative feature learning approach for deep face recognition,” in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., pp. 499–515, Springer International Publishing, Cham, 2016.
- [46] D. E. King, “Dlib-ml: a machine learning toolkit,” *Journal of Machine Learning Research*, vol. 10, pp. 1755–1758, 2009.
- [47] B. Bayar and M. C. Stamm, “A deep learning approach to universal image manipulation detection using a new convolutional layer,” in *Proceedings of the 4th ACM Workshop on Information Hiding and Multimedia Security, ser. IHMMSec*

- '16, pp. 5–10, Association for Computing Machinery, New York, NY, USA, 2016.
- [48] N. Rahmouni, V. Nozick, J. Yamagishi, and I. Echizen, “Distinguishing computer graphics from natural images using convolution neural networks,” in *Proceedings of the 2017 IEEE Workshop on Information Forensics and Security (WIFS)*, pp. 1–6, Rennes, France, December 2017.
 - [49] P. Zhou, X. Han, V. I. Morariu, and L. S. Davis, “Two-stream neural networks for tampered face detection,” in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 1831–1839, Honolulu, HI, USA, July 2017.
 - [50] Y. Nirkin, L. Wolf, Y. Keller, and T. Hassner, “Deepfake detection based on discrepancies between faces and their context,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 10, pp. 6111–6121, 2022.
 - [51] X. Yang, Y. Li, and S. Lyu, “Exposing deep fakes using inconsistent head poses,” in *Proceedings of the ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8261–8265, Brighton, UK, May 2019.
 - [52] Y. Li and S. Lyu, “Exposing deepfake videos by detecting face warping artifacts,” 2019.
 - [53] F. Matern, C. Riess, and M. Stamminger, “Exploiting visual artifacts to expose deepfakes and face manipulations,” in *Proceedings of the 2019 IEEE Winter Applications of Computer Vision Workshops (WACVW)*, pp. 83–92, Waikoloa Village, HA, USA, January 2019.
 - [54] H. H. Nguyen, F. Fang, J. Yamagishi, and I. Echizen, “Multi-task learning for detecting and segmenting manipulated facial images and videos,” in *Proceedings of the 2019 IEEE 10th International Conference on Biometrics Theory, Applications and Systems (BTAS)*, pp. 1–8, Tampa, FL, USA, September 2019.
 - [55] I. Masi, A. Killekar, R. M. Mascarenhas, S. P. Gurudatt, and W. AbdAlmageed, “Two-branch recurrent network for isolating deepfakes in videos,” in *Computer Vision - ECCV 2020*, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds., pp. 667–684, Springer International Publishing, Cham, 2020.
 - [56] H. H. Nguyen, J. Yamagishi, and I. Echizen, “Use of a capsule network to detect fake images and videos,” *arXiv*, 2019.

Research Article

DRT-Unet: A Segmentation Network for Aiding Brain Tumor Diagnosis

Shujing Li and Linguo Li 

College of Information Engineering, Fuyang Normal University, Fuyang 236041, China

Correspondence should be addressed to Linguo Li; llg-1212@163.com

Received 27 May 2022; Revised 8 August 2022; Accepted 22 August 2022; Published 9 September 2022

Academic Editor: Beijing Chen

Copyright © 2022 Shujing Li and Linguo Li. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Using image segmentation techniques to assist physicians in brain tumor diagnosis is a hot issue in computer technology research. Although most brain tumor segmentation networks to date have been based on U-Net, the prediction results are depending on which are not well generalized and need to be further improved. As the depth of the network increases, the gradients of the network vanish together with the decrease of the accuracy; meanwhile, the large number of parameters in the network will cause data redundancy. Moreover, a single modality of MRI images cannot adequately segment tumor details. Therefore, a segmentation network with an improved U-Net model is proposed in this paper, which combines Dilated Convolution-Dense Block-Transformation Convolution-Unet (hereafter referred to as DRT-Unet). The network adopts the combination of dilated convolution, dense residual block, and transposed convolution. In the coding process, a dilated convolution block and a local feature residual for fusing dense block are adopted to replace the 3×3 convolution layers on each layer in U-Net, and a transition layer is used for down-sampling. In the decoding process, a local feature residual is adopted for fusing dense blocks; meanwhile, a deconvolution structure with up-pooling and transposed convolution cascade is used. By connecting the decoded output features with the encoded low-level visual features, the information on transition layer loss is obtained. The experiments in this paper are carried out on BraTs2018 and BraTs2019 datasets; as a result, the DRT-Unet network can effectively segment tumor lesion regions.

1. Introduction

Brain tumors are a general term for tumors of the nervous system that grow inside the skull, second only to tumors of the lung, stomach, uterus, and breast, accounting for approximately 5% of systemic tumors, 70% of childhood tumors [1], and more than 2.4% of deaths [2]. Magnetic resonance imaging (MRI) is one of the most commonly used diagnostic techniques in clinical care, which is particularly important in the diagnosis of brain tumors. It is noninvasive, accurately providing the shape, size, and location of the brain tumors without the patients receiving high ionizing radiation, as well as having good soft tissue contrast [3]. Accurate segmentation of brain tumors is of essential importance for disease diagnosis, pathological research, and later surgical plan determination.

Although accurate segmentation of brain tumors is required in clinical research, it is usually filled with challenges, mainly including image artifacts, noise, and low contrast, as well as considerable variations in tumor shape, size, and location from case to case. What is more, the segmentation of brain tumors is more challenging since the boundaries between the structures of brain tumors are fuzzy and the internal structures are similar. Manual segmentation of the brain tumors, which depends on the doctor's expertise and experience, is quite cumbersome. Therefore, the study of a method that can automatically, accurately, and effectively segment brain tumors is of great significance for clinical diagnosis and surgery.

In recent years, deep learning has achieved a series of successes in many fields such as image, audio, and natural language. Among them, convolutional neural networks

(CNNs) have important references in computer vision tasks [4–9], and significant progress has been made in semantic image segmentation [10, 11]. CNNs learn visual and semantic features in images during the training process, reducing the complexity of the network model and making it possible to train networks in depth. In summary, deep CNNs have a wide range of applications in medical image processing [12–14]. According to the different input and output methods, the image segmentation method based on deep learning can be divided into block segmentation and end-to-end segmentation, and the latter is mainly realized through the encoder-decoder structure. The complete image or image block is input, and the type probability of each pixel in the output image is decoded, so as to achieve the purpose of tumor region segmentation. The relevant model of this method is mainly based on the U-Net network, which proposes a symmetric structure with jump connections to retain image details, becoming the mainstream framework for most image segmentation tasks. Although the improved method based on U-Net improves the segmentation performance, there is still room for improvement in network depth and generalization. In recent years, the concept of identity mapping has been introduced to balance the depth and performance of the network. However, the use of residual blocks to adjust the number of channels makes the number of channels increase dramatically, resulting in data redundancy. Unimodal MRI images cannot complete the full segmentation of tumor-related areas and details, and the use of multimodal brain tumor images can make up for the above weaknesses.

In this paper, the characteristics and performance of each model are combined and a multimodal brain tumor segmentation method is proposed for the DRT-Unet network, which is similar to U-Net in the overall framework. The exact contributions of this paper are as follows:

- (1) The ordinary convolution with dilated convolution is combined to expand the sensory field and optimize the feature extraction capability. While introducing two mapping methods, 3×3 ordinary convolution and dilated convolution in parallel can obtain a sensory field larger than 9 frames, with a greater sensory field and better feature extraction capability. Each pixel in the output feature map can respond to a larger area in the image.
- (2) The dense block used in this paper consists of a dense layer and a residual fusion of local features. The “jump connection” of ResNets is introduced in the down-sampling process, which is combined with the dense block to preserve and propagate the rich low-level visual features [15], such as brain tumor brightness, color, texture, and other features that directly stimulate vision.
- (3) In the decoding process, dense blocks are fused by using local feature residuals to form a cascaded deconvolution structure, so that the output image has the same dimension as the input image, while in the decoding process, low-level visual features from

the encoding process are connected with the same dimension and channels; meanwhile, features are fused to obtain the missing information after the transition layer in the encoding path.

2. Related Work

Currently, most deep CNNs used for brain tumor segmentation networks are end-to-end. End-to-end brain tumor segmentation networks use an encoding-decoding approach where the input is the whole image or image block. The features are extracted by encoding in the convolutional layer, which is then decoded to obtain the class probability of each pixel point in the whole image or image block finally. Such a segmentation method is mainly based on FCN [16] and U-Net networks [17]. Raza et al. [18] proposed a hybrid model based on a deep residual network and U-Net, which takes the residual network as the encoder to deal with the problem of gradient disappearance, as well as uses low-level and high-level features to predict. Nevertheless, this method ignores the context information, resulting in high computational costs. Zhang et al. [19] proposed a multi-scale mesh aggregation network. By introducing an improved inception module to replace the standard convolution, effective information is extracted and aggregated from different receptive fields, and the network aggregation strategy is adopted to gradually refine shallow features. However, the number of network parameters is large, accompanied by low segmentation efficiency. Chen et al. [20] proposed a symmetric network based on a deep convolution neural network, which expanded the functional mapping between low-level and high-level features by adding symmetric masks in multiple layers, and combined the prior knowledge of symmetry with brain tumor segmentation; however, the effect of low-contrast tumor segmentation was poor. Wang et al. [21] proposed a segmentation network based on a segmented attention module, which extracted useful information in connected features through different attention mechanisms and discarded redundant information to realize selective aggregation of features. What is more, Wang et al. [22] proposed extracting multi-scale image features by using a spatial module composed of multiple parallel dilated convolution layers and deepening the network structure by using a residual module. Shen et al. [23] proposed a multi-task full convolutional network for the automatic segmentation of brain tumors. Based on the hierarchical relationship between tumor substructures, the network takes multimodal MRI images and their symmetric differential images as inputs to extract multi-level background information. Experiments showed that the proposed multi-task FCN outperformed single-task FCN for all subtasks. However, there were limitations in the FCN-based approach for predicting low-resolution images [24]. Based on U-Net network, Hao et al. [25] proposed a new network for brain tumor segmentation. They used a comprehensive data enhancement scheme to preprocess the data and conducted the experiments with BraTs2015 dataset. The DSC values obtained in the intact tumor region, the core tumor region, and the enhanced tumor region were 0.86, 0.86, and 0.65,

respectively. Although the performance of the network can be improved as more layers are added to the network, degradation and gradient disappearance can occur with the deepening of the network. The ResNets proposed by He et al. [26] in 2015 cope with this problem by introducing the concept of residual blocks. ResNets perform well in a range of image recognition, localization, and detection tasks, such as ImageNet and COCO object detection. The literature [24] proposes RefineNet, a generalized multi-path optimization network whose components are connected by using residuals according to the idea of identity mapping, so as to achieve efficient end-to-end training. Experimentally, the method proved to improve the performance of the segmentation. However, the addition of a multi-path refinement network on ResNets increased the parameters of that network as well.

DenseNet [27], as the best paper of CVPR2017, does not take deepening and widening the network as the two ways to improve the performance of the network, but considers the feature perspective, greatly reducing the number of parameters as well as alleviating the problem of gradient disappearance through feature reuse and bypass settings. The authors in the literature [28] proposed a fully convolutional network for semantic segmentation, i.e., FC-DenseNet, by fusing dense blocks in DenseNet with jump connections of ResNets. Kaku et al. [29] proposed a brain tumor segmentation network named DenseUnet by incorporating the dense block structure into U-Net and conducted experiments in Mindboggle-101 and New York University (NYU) artificial correction dataset. The best Dice values of 0.819 ± 0.011 and 0.800 ± 0.012 were obtained, respectively, which were better than the segmentation performance of U-Net.

3. Network Model

On the basis of the advantages and disadvantages of FCN, U-Net, ResNets, and DenseNet and the computational principles of CNNs, this paper proposes a segmentation network of DRT-Unet, whose network structure is shown in Figure 1. The DRT-Unet network is similar to U-Net in the overall framework; meanwhile, a dense fusion of dilated convolution blocks, as well as local feature residuals, is used in the encoding process. In the coding process, the dilated convolution block and the local feature residual fusion dense block are used instead of two repetitive 3×3 convolution layers in U-Net: the dilated convolution block consists of dilated convolution and normal convolution, which can expand the perceptual field without losing local information. An $X \times X$ convolution layer can make the value of each pixel feel an area of X^2 size; for example, a 3×3 convolution layer can obtain the receptive field of 9 lattice size, but the parallel connection of ordinary convolution and dilated convolution can not only obtain receptive fields larger than 9 lattices, but also introduce two mapping methods at the same time. Therefore, the combined hole-convolutional block has larger receptive fields and better feature extraction ability. The dense block in this paper is composed of a dense layer and residual fusion of local features, and the identity mapping of

ResNet is connected to the dense block and the coding process, so as to retain and spread more low-level visual features. The transition layer is used for down-sampling in the coding process. The decoding process is implemented by using a deconvolution structure with local feature residual fusion dense blocks and up-pooling and transposed convolution cascades, while the decoding process connects with the lower-level visual characteristics in the encoding process with the same dimension and number of channels, and the features with high-level semantic information are then fused to generate new features to obtain the missing information after the transition layer in the encoding path. (Since then, the deconvolution in the following refers to the cascading operation of up-pooling and transposed convolution in the decoding network).

It is required that the feature maps remain the same size in the same dense block, so only a transition layer between different dense blocks is implemented for down-sampling. To further decrease the network parameters, a 1×1 convolution operation is inserted between every two dense blocks. The transition layer is the TD (transition down) module in Figure 1, whose specific structure is BN + Conv (1×1) + 2×2 max-pooling. The number of layers in the down-sampling part of the network is set according to the number of layers in the first four dense blocks in FC-DenseNet.

3.1. Dilated Convolution Block. Deep CNNs usually use down-sampling or convolutional layers to enhance the perceptual field of the network, which, however, will reduce the spatial resolution. In order to be able to balance both resolution and perceptual field, the literature [30] proposes dilated convolution, also known as dilated convolution. Therefore, in this paper, the coding process of the network uses the dilated convolution at each layer instead of the twice repeated 3×3 convolution used in the U-Net network.

The computational effort of the dilated convolution is comparable to that of the conventional convolution, except that the sampling density of data is changed. However, in the results of a layer obtained by using the dilated convolution, the neighboring pixels are obtained from the convolution of independent subsets. There is a lack of correlation between them as well as a problem of local information loss, while the network model in this paper adopts the dilated convolution and normal convolution in parallel to obtain this information and solve the problem of lack of correlation between the convolution results. Input image or image block, a 3×3 dilated convolution layer, and a 3×3 ordinary convolution layer are used to extract features under different receptive fields, and the feature extraction results under two different mapping methods are obtained. Two different feature extraction result matrices are spliced, and the combined hole convolution block output results are obtained through the activation function. Compared with 3×3 ordinary convolution, the result has a larger receptive field and richer contextual information. The structure of the dilated convolution block used in this paper is shown in Figure 2.

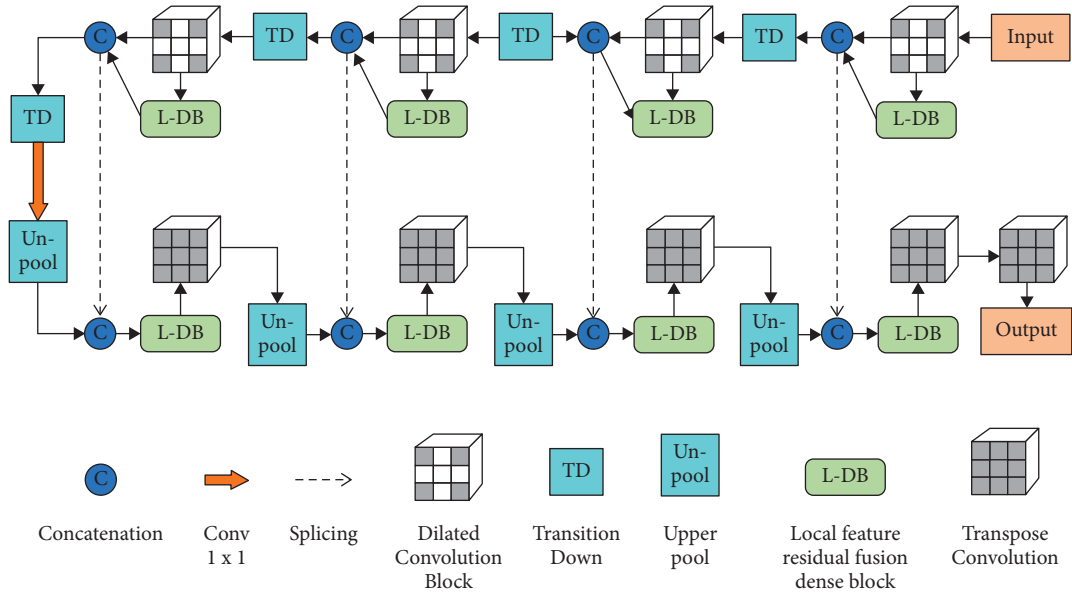


FIGURE 1: DRT-Unet structure diagram.

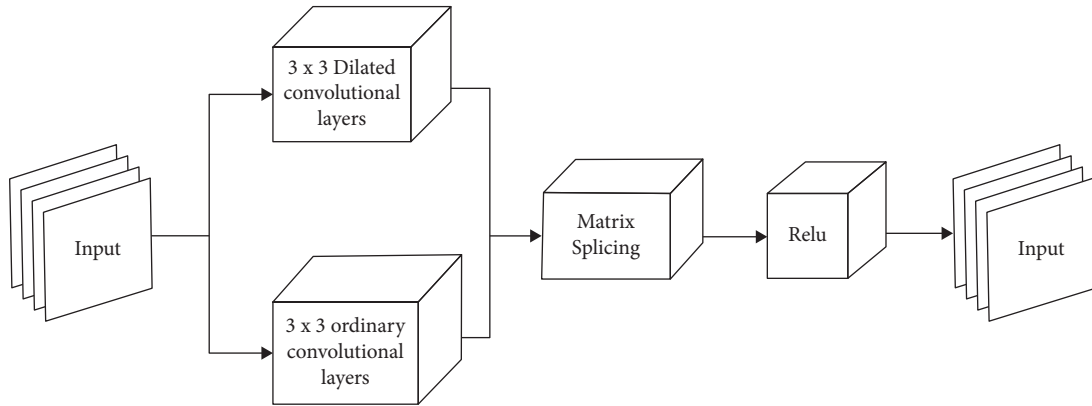


FIGURE 2: Structure of the dilated convolution block.

3.2. Dense Residual Block. Recent research has shown that CNNs can be trained more deeply, accurately, and efficiently if shorter connections are used between their input and output layers. Based on this conclusion, the literature [27] proposed DenseNets, which consist of dense blocks that connect each layer with all previous layers. The features on the input of each of these layers are the outputs of all previous layers, and the features on the output of each layer will be used as the input of all subsequent layers. This dense connection now makes a direct connection on the input and loss at each layer, so the dense network can mitigate the problem of gradient disappearance. Given the above advantages of dense blocks, the dense block of local feature residual fusion from the literature [15] is cited in this paper, which consists of two parts, i.e., a densely connected block and a local feature residual block fusion.

3.3. Deconvolutional Network. The DRT-Unet network uses local feature residuals to fuse dense blocks and up-pooling and transposed convolutional cascade with a deconvolution

structure to realize the decoding process of feature map size scaling. The information generated by the dense block during the encoding process is lost after the transition layer, but this lost information can be obtained in the decoding path by making a jump connection to the encoding path. Thus, the feature map after the up-pooling operation is jump-connected with the features of the same layer in the coding network to form the input of the next dense block. In order to reduce the spatial dimension, the input of the dense block in the deconvolution-decoding network is not cascaded with its output. As shown in Figure 1, the coding network in the upper half is a feature extractor that extracts feature descriptions from the input image, while the decoding network in the lower half is a shape generator used to generate segmentation targets from the extracted feature maps. It can be seen that the deconvolution-decoding network is almost a mirror result of the convolution-encoding network.

The feature map recovered by up-pooling becomes a sparse feature map due to the presence of a large number of 0 elements. The transposed convolution refers to the

transposition of the convolution kernel learned in the process of convolution. Compared with the sparse feature map obtained after up-pooling, this sparse feature map is used to form a dense feature map so as to correspond multiple feature maps to one feature map. In order to maintain the same size as the feature map obtained from the up-pooling, the obtained dense feature map needs to be cropped accordingly. Using the convolution kernel learned by transposed convolution and up-pooling, the shape of the target object based on the reconstruction is obtained in the deconvolution network. By applying the structure of up-pooling and transposed convolutional cascades in the decoding path and combining jump connections to compensate for the missing information, a deeper dense block network is constructed without generating a feature map explosion.

4. Experimental Results and Analysis

4.1. Dataset and Preprocessing. The datasets used in this paper were derived from BraTs2018 and BraTs2019, in which each case has four modalities, namely, T1-weighted imaging, T2-weighted imaging, contrast-enhanced T1ce, and liquid-attenuated inversion recovery column Flair [31], with different imaging modalities providing different information about brain tumors (each modality represents a different response to different tumor tissues) [32]. Although MRI images can quickly and effectively detect changes in water content in the sensing region and provide rich diagnostic information, a single MRI modality image cannot adequately subdivide the tumor in the region of interest and therefore cannot solve the problem of precise regional segmentation. Besides, using different MRI modalities can compensate for the above weaknesses. Hence, the slices of four modalities are used in this paper as the input of the segmentation network.

In this paper, BraTs2018 is selected as the training set, which contains 285 cases. Among them, 210 cases of HGG and 75 cases of LGG are included [33]. BraTs2019 has added 49 cases of HGG and 1 case of LGG based on BraTs2018, the new addition of which is used as the validation set. The dimension of each MRI image in the dataset is $155 \times 240 \times 240$. MRI images are represented as stereoscopic pixels in NIFTI format, and a series of preprocessing operations are required to fit the 2D network in this paper. A dichotomous segmentation was used to cut the brain tumor cases from the cross-sectional plane and obtain 2D images. A z-score approach is then used to normalize each modal image [34]. To alleviate the inter-category imbalance problem, slices without lesions in the image are discarded. To enhance the performance of model segmentation, the original 2D images are cropped from width and height dimensions of 240 mm to 160 mm in this paper, and the linear features and corresponding distribution relations of the image distribution are not changed during cropping. After the above steps, the images were divided into 4 channels and saved as an array of formats for data training and validation. Finally, the training set contains 17,925 slices and the validation set contains 7,750 slices.

4.2. Evaluation Indicators. In order to measure the effect of DRT-Unet network on brain tumor segmentation in a comprehensive and multi-faceted way, this paper adopts Dice [35], positive predictive value (PPV), sensitivity, intersection over union (IoU) ratio [36], and Hausdorff distance (95% HD) [37] as evaluation indexes, and the prediction results are compared with the real labeled data to show the segmentation effect from a visual perspective.

Dice is used to measure the resemblance between the segmentation result and the true value. The value of Dice is 1 when the segmentation result is best, and 0 when it is worst, which is defined by the following formula:

$$\text{Dice}\langle A, B \rangle = 2 \times \frac{|A \cap B|}{(|A| + |B|)} = \frac{2TP}{FP + 2TP + FN}. \quad (1)$$

PPV indicates the proportion of samples with positive predictions that are correctly predicted, and it is defined by the following formula:

$$\text{PPV} = \frac{TP}{TP + FP}. \quad (2)$$

Sensitivity indicates the proportion of samples predicted to be positive to the total positive samples (true-positive rate), which is defined by the following formula:

$$\text{Sensitivity} = \frac{TP}{TP + FN}. \quad (3)$$

IoU is a measure of the accuracy of the detection object, which is defined by the following formula:

$$\text{IoU} = \frac{TP}{FP + TP + FN}. \quad (4)$$

The Hausdorff distance (95% HD) is sensitive to contour information. The more this value tends to 0, the more accurate the predicted value is. In order to exclude the instability and unreasonableness of the segmented data caused by a few outliers, the parameter 95% was chosen as the maximum distance quartile, which is defined as follows:

$$\begin{aligned} 95\%HD &= \max\{d_{XY}, d_{YX}\} \\ &= \max \left\{ \max_{x \in X} \min_{y \in Y} \left\{ \max_{y \in Y} \min_{x \in X} \right\} \right\}, \end{aligned} \quad (5)$$

where A and B are the actual expert data values and model prediction values, respectively. TP indicates a positive sample with a positive model prediction, TN indicates a negative sample with a negative model prediction, FP represents a negative sample with a positive model prediction, and FN denotes a positive sample with a negative model prediction.

4.3. Experimental Parameter Settings. In this paper, PyTorch library is adopted to build DRT-Unet network, while Adam optimizer is used to train the method, with the training batch

being 20 and the training round being 400. The initial learning rate is set to 2×10^{-4} , with the weight decay coefficient set to 0.0002. k in the dense block is an indicator of the number of feature map output per layer in each dense block, which is set differently in this paper. Since Dice is more commonly used than the other three evaluation metrics, only the average value of Dice over the three segmentation regions is used as the reference standard in the stage of determining the parameter k . The results of model segmentation under different k are shown in Table 1, where the optimal data are in bold.

The best segmentation results are obtained when the value of k is 16. The experimental results show that the smaller the k is, the better the segmentation effect is; meanwhile, the network can be avoided to become too wide.

4.4. Experimental Results and Discussion. In order to verify the impact of each module in DRT-Unet network on the segmentation performance, this paper takes the U-Net network model as the basis, then adds the local feature residual fusion dense block and the dilated convolution in turn, respectively, to improve it, and finally, compares the obtained experimental results with DRT-Unet. As shown in Table 2, the experimental results in the table are mean values, where the optimal data are in bold.

U-Net, as the reference basic framework in this paper, obtained Dice coefficient of 0.810, precision of 0.822, recall of 0.919, and 95% HD of 1.157. After replacing the two 3×3 convolutions on each layer in the U-Net coding process with a dense block of local feature residual fusion, all four metrics are improved, with precision improved by 2.8% over the U-Net network, which indicates that the dense block of local feature residual fusion can effectively propagate and retain low-level visual features, and can reduce the information loss in deep network training through the fusion of local features. When the dilated convolution block is added to the encoding process, precision and recall are improved by 1.7% and 1.6%, respectively, based on the previous step, indicating that the fusion of normal convolution and dilated convolution can expand the perceptual field, obtain richer features, and provide more detailed information. As can be seen from the data in Table 2, the values of Dice and 95% HD have significantly changed, increasing by 2.2% and decreasing by 2.9%, respectively, compared with the previous method, which indicates that the combination of up-pooling and transposed convolution can effectively capture the global features and detailed features, and recover the extracted features well to the original pixels. Finally, the Dice value of DRT-Unet is 0.861, the precision value is 0.881, the recall value is 0.948, and the 95% HD value is 1.112. Compared with the U-Net network, these four metrics are improved by 5.1%, 5.9%, 2.9%, and 5.3%, respectively, which fully demonstrates the effectiveness of the proposed method in this paper.

In order to further prove the segmentation performance of this method, the classical deep learning segmentation networks FCN8s, U-Net, and the methods in literature [29] (DenseUnet), literature [38] (DeepResUnet), literature [39]

TABLE 1: Dice coefficient values at different k .

k	Dice coefficient value
16	0.861
32	0.796
48	0.789

TABLE 2: Comparison of different segmentation network models in each index.

Network model	Dice	Precision	Recall	95% HD
U-net	0.810	0.822	0.919	1.157
L-DB + U-Net	0.823	0.850	0.929	1.146
Dilated conv + L-DB + U-Net	0.839	0.867	0.945	1.133
DRT-Unet	0.861	0.881	0.948	1.104

(H 2 NF-Net), as well as literature [40] (MCA-ResUNet) are compared with DRT-Unet network, and all experiments use multimodal images of the same dataset as the input to the network. The goal of this paper is to segment three regions, WT, TC, and ET. WT is the intact tumor, which represents a blue region in the figure. Preoperative MRI images showing the extent and volume of the edema of the intact tumor can achieve high-precision localization of the tumor. TC is the core tumor, which corresponds to the white region in the figure and is a malignant tumor evolving from glial cells in the brain. The red region belongs to ET, which is the tumor-enhancing necrotic region, and is composed of necrotic cells. In this paper, the Flair sequence with the most obvious bright contrast is selected as the original contrast image from four modalities, T1, T2, Flair, and T1ce. Three cases with different characteristics are selected for doing visual effect comparison, the results of which are shown in Figures 3–5. In the figures, (a) is the original Flair sequence image, (b) is the manual segmentation label, and (c), (d), (e), (f), (g), (h), and (i) are the segmentation results of FCN8s, U-Net, DenseUnet, DeepResUnet, H 2 NF-Net, MCA-ResUNet, and DRT-Unet, respectively.

From the experimental results of the above three cases, the segmentation results of FCN8s are relatively rough. Figures 3 and 4 clearly reveal that the segmentation at the edge of the tumor is unclear, and the TC and ET regions cannot be finely segmented, with the poor overall effect. DenseUnet uses dense blocks to compensate for the detailed information of U-Net, which substantially improves the segmentation effect. However, by observing Figure 3, we find that the segmentation of WT edge region appears hollow. In Figure 5, we find that the segmented edge branching region is broken, and the contour is incoherently connected. DeepResUnet uses residual blocks to fuse multidimensional features. Although the segmentation results are generally better, the generalization ability is poor and there are many fragmented points at the boundary of WT region. Compared with U-Net segmentation results, DRT-Unet segmentation in WT area is more accurate, and the false segmentation region is smaller. It is obviously shown in Figure 4 that U-Net is unable to perform fine segmentation for areas with irregular edges, while the outline of DRT-Unet is closer to manual labels, showing that the dilated convolution block

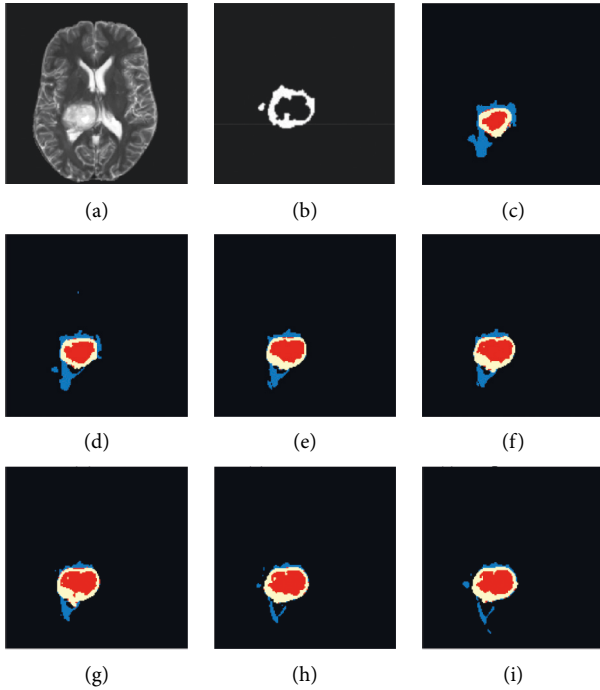


FIGURE 3: Case 1: weak contrast between light and dark inside the image, and difficulty in further fine segmentation of heterogeneous regions. (a) Original image. (b) Handmade labels. (c) FCN8s. (d) U-Net. (e) DenseNet. (f) DeepResUNet. (g) H 2 NF-Net. (h) MCA-ResUNet. (i) DRT-Unet.

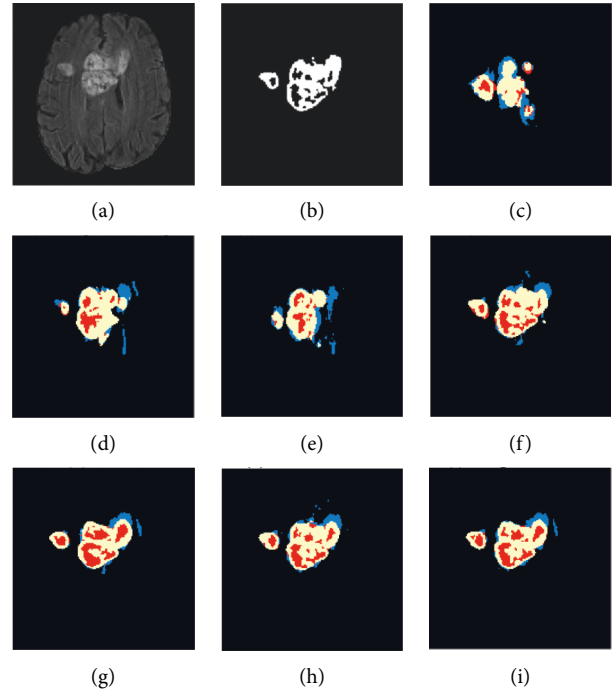


FIGURE 5: Case 3: the image is characterized by complex edges and many contour bifurcations, which can reflect the segmentation of contour details. (a) Original image. (b) Handmade labels. (c) FCN8s. (d) U-Net. (e) DenseNet. (f) DeepResUNet. (g) H 2 NF-Net. (h) MCA-ResUNet. (i) DRT-Unet.

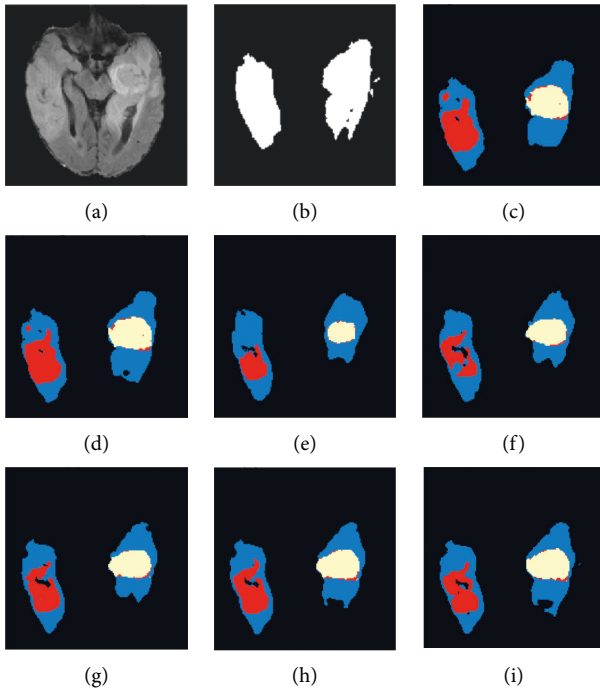


FIGURE 4: Case 2: tumor spread and metastasis can reflect segmentation effects on multiple regions. (a) Original image. (b) Handmade labels. (c) FCN8s. (d) U-Net. (e) DenseNet. (f) DeepResUNet. (g) H 2 NF-Net. (h) MCA-ResUNet. (i) DRT-Unet.

can obtain more abundant features; in addition, the combination of up-pooling and transposed convolution can effectively capture detailed features. DRT-Unet segmentation in each region is more complete compared with the above methods, the contrast between the core tumor region and the intact tumor region is clearer, and the contour line segmentation also performs better in detail than the existing algorithms.

Meanwhile, the IoU curves and loss function curves of the proposed methods in this paper are compared with those of FCN8s, FCN16s, FCN32s, U-Net, DenseUNet, and DeepResUNet, as shown in Figures 6 and 7. From these two figures, it can be seen that the IoU value of DRT-Unet is significantly higher than that of other methods, while the final loss function value is the lowest.

To better reflect the segmentation effect, the segmentation results of WT, TC, and ET were further and quantitatively analyzed by four assessment metrics, namely, Dice, precision, recall, and 95% HD, respectively. The whole tumor (WT) category includes all visible labels (a union of blue, yellow, and red labels), while the tumor core (TC) category is a union of red and yellow. Different from the two mentioned above, the enhancing tumor (ET) core category is only yellow (a hyperactive tumor part). The comparison results between DRT-Unet and other networks are shown in Tables 3–6, where the optimal data are shown in bold. As can

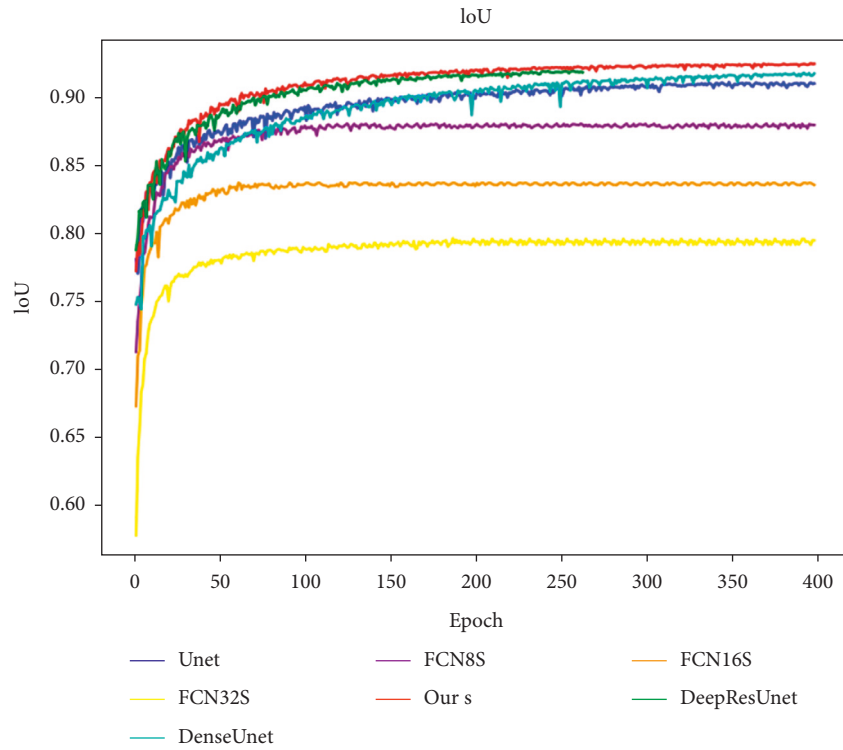


FIGURE 6: IoU curve comparison.

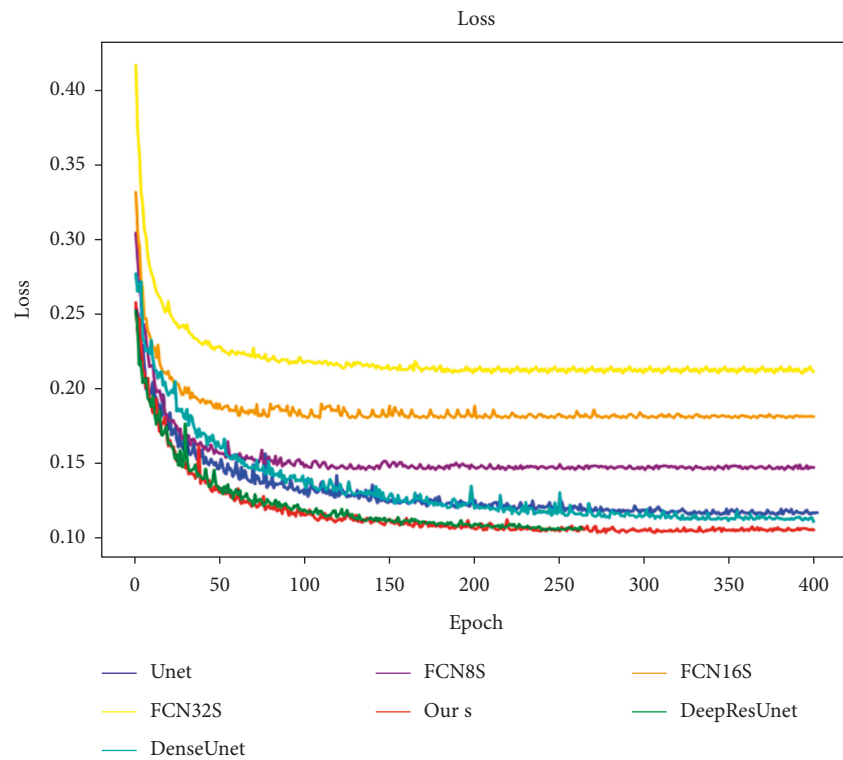


FIGURE 7: Comparison of loss function curves.

be seen from the table, the best data of DRT-Unet on these four evaluation indicators can reach 0.918, 0.93, 0.968, and 0.748, respectively. Besides, the values of each index on TC

are higher than those on WT and ET regions, which indicates that the TC region is relatively well segmented. Although the precision and Dice of DRT-Unet are slightly worse than

TABLE 3: Dice coefficient values under different networks.

Dice	Network								
	FCN8s	FCN16s	FCN32s	U-Net	DenseUnet	DeepResUnet	H2 NF-net	MCA-ResUNet	DRT-Unet
WT	0.564	0.539	0.418	0.769	0.792	0.799	0.913	0.849	0.842
TC	0.897	0.898	0.872	0.917	0.921	0.92	0.849	0.865	0.918
ET	0.531	0.495	0.351	0.743	0.765	0.765	0.79	0.784	0.823
Average	0.664	0.644	0.547	0.81	0.826	0.828	0.85	0.833	0.861

TABLE 4: Precision values under different networks.

Precision	Network								
	FCN8s	FCN16s	FCN32s	U-Net	DenseUnet	DeepResUnet	H2 NF-Net	MCA-ResUNet	DRT-Unet
WT	0.564	0.536	0.414	0.783	0.809	0.812	0.83	0.868	0.842
TC	0.911	0.905	0.897	0.935	0.932	0.926	0.912	0.89	0.918
ET	0.526	0.485	0.357	0.76	0.782	0.791	0.788	0.805	0.823
Average	0.667	0.642	0.556	0.826	0.841	0.843	0.854	0.854	0.861

TABLE 5: Recall values under different networks.

Recall	Network								
	FCN8s	FCN16s	FCN32s	U-Net	DenseUnet	DeepResUnet	H2 NF-Net	MCA-ResUNet	DRT-Unet
WT	0.94	0.913	0.91	0.925	0.928	0.94	0.923	0.86	0.95
TC	0.918	0.926	0.922	0.935	0.966	0.966	0.9	0.845	0.968
ET	0.908	0.879	0.853	0.909	0.914	0.915	0.894	0.917	0.923
Average	0.922	0.906	0.895	0.923	0.936	0.942	0.915	0.874	0.947

TABLE 6: 95% HD values under different networks.

95% HD	Network								
	FCN8s	FCN16s	FCN32s	U-Net	DenseUnet	DeepResUnet	H2 NF-Net	MCA-ResUNet	DRT-Unet
WT	1.809	2.211	2.598	1.334	1.42	1.345	1.1	2.592	1.252
TC	0.788	0.829	0.913	0.711	0.698	0.749	1.14	1.595	0.748
ET	2.114	2.27	2.738	1.417	1.476	1.407	2.61	2.739	1.315
Average	1.579	1.77	2.083	1.154	1.198	1.167	1.617	2.31	1.105

DenseUnet in the TC region, all the metrics of WT and ET are improved to various degrees compared with other methods, and the mean values of all the metrics of the proposed method are the highest in all three regions, indicating that DRT-Unet can segment WT, TC, and ET better and achieve more satisfactory segmentation results.

5. Conclusion

At present, the majority of brain tumor segmentation methods are based on two networks, FCN and U-Net, but the network connection based on FCN is not fine-grained and ignores the relationship between different pixel points. The U-Net model is experimentally proven to be slightly improved compared with FCN, but the overall generalization of prediction results is not strong and needs to be improved to a certain depth. To address these problems, this paper proposes a DRT-Unet network for the accurate segmentation of brain tumors, where four MRI modality images are used as input, and a dilated convolution block is used to expand the perceptual field in the coding process, so that the network can obtain richer and more detailed

feature information. Meanwhile, a dense block of local feature residual fusion is used in the coding process to propagate and preserve low-level visual features, reducing the information loss in deep network training through the fusion of local features. The DRT-Unet network adopts a dense block of local feature residual fusion and a deconvolution structure of up-pooling and transposed convolution cascade to achieve the decoding process of feature map size enlargement. The up-pooling and transposed convolution play a key role in recovering the global features and detailed features of the image. It can be seen from the experiments in this paper and the comparison with other methods that the DRT-Unet method can achieve effective segmentation of brain tumor lesions. Moreover, compared with the other four segmentation methods, the proposed network in this paper has better performance in visual effects and objective indexes.

Data Availability

The datasets used in this paper were derived from BraTs2018 and BraTs2019, and these are open-access data.

Conflicts of Interest

The authors declare that they have no conflicts of interest with any organization or entity in this manuscript.

Acknowledgments

This work was supported in part by the National Youth Natural Science Foundation of China (61802208), the Natural Science Foundation of Anhui (1908085MF207, KJ2020A1215, KJ2020A1216, and KJ2021A1251), the Excellent Youth Talent Foundation of Anhui (gxyqZD2019097 and gxyqZD2021142), the Postdoctoral Foundation of Jiangsu (2018K009B), the Quality Engineering Project of Anhui (2021jyxm1117, 2021kcszsfkc307, 2018mooc059, 2018kfk009, 2018sxzx38, 2018FXJT02, and 2018sxzx38), the Fuyang Normal University Doctoral Startup Foundation (2017KYQD0008), the Industry-University Cooperation Collaborative Education Project (201901258002).

References

- [1] T. X. Gao, Z. Lv, H. Y. Ding, and X. Y. Tang, "Research advances in magnetic resonance temperature imaging," *Chinese Journal of Medical Imaging*, vol. 22, no. 7, pp. 547–550, 2014.
- [2] R. Siegel, D. Naishadham, and A. Jemal, "Cancer statistics," *CA: A Cancer Journal for Clinicians*, vol. 62, no. 1, pp. 10–29, 2012.
- [3] P. L. Zhi and P. C. Lauterbur, "Principles of magnetic resonance imaging: a signal processing perspective," *Spie Optical Engineering*, Wiley-IEEE Press, Hoboken, NJ, USA, 2000.
- [4] W. Li, S. Cheng, K. Qian, K. Yue, and H. Liu, "Automatic recognition and classification system of thyroid nodules in CT images based on CNN," *Computational Intelligence and Neuroscience*, vol. 2021, pp. 1–11, Article ID 5540186, 2021.
- [5] Z. Qin, Y. Wang, H. Cheng, Y. Zhou, Z. Sheng, and V. C. M. Leung, "Demographic information prediction: a portrait of smartphone application users," *IEEE Transactions on Emerging Topics in Computing*, vol. 6, no. 3, pp. 432–444, 2018.
- [6] Z. Qin, Y. Zhang, S. Meng, Z. Qin, and K. K. R. Choo, "Imaging and fusing time series for wearable sensor-based human activity recognition," *Information Fusion*, vol. 53, pp. 80–87, 2020.
- [7] B. Xiao, K. Wang, X. Bi, W. Li, and J. Han, "2D-LBP: an enhanced local binary feature for texture image classification," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, p. 1, 2018.
- [8] H. Tang, B. Xiao, W. Li, and G. Wang, "Pixel convolutional neural network for multi-focus image fusion," *Information Sciences*, vol. 433–434, pp. 125–141, 2018.
- [9] B. Xiao, Y. Xu, X. Bi et al., "Follow the sound of children's heart: a deep learning-based computer-aided pediatric CHDs diagnosis system," *IEEE Internet of Things Journal*, vol. 7, no. 3, pp. 1994–2004, 2020.
- [10] Y. Ding, F. Chen, Y. Zhao, Z. Wu, C. Zhang, and D. Wu, "A stacked multi-connection simple reducing net for brain tumor segmentation," *IEEE Access*, vol. 7, pp. 104011–104024, 2019.
- [11] Y. Ding, C. Luo, C. Li, T. Lan, and Z. G. Qin, "High-order correlation detecting in features for diagnosis of Alzheimer's disease and mild cognitive impairment," *Biomedical Signal Processing and Control*, vol. 53, pp. 101564–101571, Article ID 101564, 2019.
- [12] M. Havaei, A. Davy, D. Warde-Farley et al., "Brain tumor segmentation with deep neural networks," *Medical Image Analysis*, vol. 35, pp. 18–31, 2017.
- [13] K. Kamnitsas, C. Ledig, V. F. J. Newcombe et al., "Efficient multi-scale 3D CNN with fully connected CRF for accurate brain lesion segmentation," *Medical Image Analysis*, vol. 36, pp. 61–78, 2017.
- [14] B. Kayalibay, G. Jensen, and V. D. S. Patrick, "CNN-Based segmentation of medical imaging data," 2017, <https://arxiv.org/abs/1701.03056>.
- [15] Y. Ding, L. Gong, M. Zhan, C. Li, and Z. Qin, "A multi-path adaptive fusion network for multimodal brain tumor segmentation," *Neurocomputing*, vol. 412, no. 1, pp. 19–30, 2020.
- [16] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 640–651, 2017.
- [17] O. Ronneberger, F. Philipp, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proceedings of the 2015 International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241, Springer, Munich, Germany, October 2015.
- [18] R. Raza, U. I. Bajwa, Y. Mehmood, M. W. Anwar, and M. H. Jamal, "dResU-Net: 3D Deep Residual U-Net Based Brain Tumor Segmentation from Multimodal MRI," *Biomedical Signal Processing and Control*, Article ID 103861, 2022.
- [19] Y. Zhang, Y. Lu, W. K. Chen, Y. K. Chang, H. M. Gu, and B. Yu, "MSMANet: a multi-scale mesh aggregation network for brain tumor segmentation," *Applied Soft Computing*, vol. 110, Article ID 107733, 2021.
- [20] H. Chen, Z. G. Qin, Y. Ding, L. Tian, and Z. Qin, "Brain tumor segmentation with deep convolutional symmetric neural network," *Neurocomputing*, vol. 392, pp. 305–313, 2020.
- [21] Y. L. Wang, Z. J. Zhao, S. Y. Hu, and F. L. Chang, "CLCU-Net: cross-level connected U-shaped network with selective feature aggregation attention module for brain tumor segmentation," *Computer Methods and Programs in Biomedicine*, vol. 207, no. 106154, 2021.
- [22] J. J. Wang, J. Gao, J. W. Ren et al., "DFP-ResUNet: Convolutional neural network with a dilated convolutional feature pyramid for multimodal brain tumor segmentation," *Computer Methods and Programs in Biomedicine*, vol. 208, Article ID 106208, 2021.
- [23] H. Shen, R. Wang, J. Zhang, and S. Mckenna, "Multi-task fully convolutional network for brain tumour segmentation," in *proceedings of the Annual Conference on Medical Image Understanding and Analysis*, pp. 239–248, Edinburgh, UK, July 2017.
- [24] G. Lin, M. Anton, C. Shen, and I. Reid, "RefineNet: multi-path refinement networks for high-resolution semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1925–1934, San Juan, PR, USA, June 2017.
- [25] D. Hao, Y. Guang, F. Liu, Y. Mo, and Y. Guo, "Automatic brain tumor detection and segmentation using U-net based fully convolutional networks," in *proceedings of the Annual Conference on Medical Image Understanding and Analysis*, pp. 506–517, Edinburgh, UK, July 2017.
- [26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference*

- on *Computer Vision and Pattern Recognition*, pp. 770–778, San Juan, PR, USA, July 2016.
- [27] G. Huang, L. Zhuang, V. D. M. Laurens, and K. Q. Weinberger, “Densely connected convolutional networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4700–4708, Honolulu, HI, USA, July 2017.
- [28] S. Jégou, D. Michal, D. Vazquez, A. Romero, and Y. Bengio, “The one hundred layers tiramisu: fully convolutional DenseNets for semantic segmentation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 11–19, Honolulu, HI, USA, July 2017.
- [29] A. Kaku, C. Hegde, J. Huang et al., “DARTS: DenseUnet-based automatic rapid tool for brain segmentation,” 2019, <https://arxiv.org/abs/1911.05567>.
- [30] JS. Rhee, “Certain radially dilated convolution and its application,” *Honam Mathematical Journal*, vol. 32, no. 1, pp. 101–112, 2010.
- [31] M. Brown, R Semelka, and T. K. Nishino, “MRI: basic principles and applications, 3rd edition,” *Medical Physics*, vol. 31, no. 1, p. 170, 2004.
- [32] L. Jin, M. Liu, J. Wang, F. Wu, T. Liu, and Y. Pan, “A survey of MRI-based brain tumor segmentation methods,” *Tsinghua Science and Technology*, vol. 19, no. 6, pp. 578–595, 2014.
- [33] A. Kermi, I. Mahmoudi, and M. T. Khadir, “Deep convolutional neural networks using U-net for automatic brain tumor segmentation in multimodal MRI volumes,” in *Proceedings of the International MICCAI Brainlesion Workshop*, pp. 37–48, Granada, Spain, September 2018.
- [34] T. Upadhaya, M. Yannick, S. Eric, P. J. L. Reste, and M. Hatt, “Prognostic value of multimodal MRI tumor features in glioblastoma multiforme using textural features analysis,” in *Proceedings of the IEEE 12th International Symposium on Biomedical Imaging (ISBI)*, Brooklyn, NY, USA, April 2015.
- [35] V. Thada and V. Jaglan, “Comparison of jaccard, dice, cosine similarity coefficient to find best fitness value for web retrieved documents using genetic algorithm,” *International Journal of Innovations in Engineering and Technology*, vol. 2, no. 4, pp. 202–205, 2013.
- [36] H. Rezaatofghi, T. Nathan, J. Y. Gwak, S. Amir, and S. Savarese, “Generalized intersection over union: a metric and a loss for bounding box regression,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 658–666, Long Beach, CA, USA, June 2019.
- [37] A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, and J. Garcia-Rodriguez, “A review on deep learning techniques applied to semantic segmentation,” 2017.
- [38] Z. Zhang, Q. Liu, and Y. Wang, “Road extraction by deep residual U-net,” *IEEE Geoscience and Remote Sensing Letters*, vol. 15, pp. 749–753, 2018.
- [39] H. Jia, W. Cai, H. Huang, and Y. Xia, “H 2 NF-net for brain tumor segmentation using multimodal MR imaging: 2nd place solution to BraTS challenge 2020 segmentation task,” *Brainlesion: glioma, multiple sclerosis, Stroke and Traumatic Brain Injuries*, pp. 58–68, 2020.
- [40] T. Cao, G. Wang, L. Ren, Y. Li, and H. Wang, “Brain tumor magnetic resonance image segmentation by a multiscale contextual attention module combined with a deep residual UNet (MCA-ResUNet),” *Physics in Medicine and Biology*, vol. 67, no. 9, Article ID 095007, 2022.

Research Article

Embedding Guided End-to-End Framework for Robust Image Watermarking

Beibei Zhang ¹, Yunqing Wu,¹ and Beijing Chen ^{1,2,3}

¹School of Computer Science, Nanjing University of Information Science and Technology, Nanjing 210044, China

²Advanced Cryptography and System Security Key Laboratory of Sichuan Province, Chengdu University of Information Technology, Chengdu 610225, China

³Jiangsu Collaborative Innovation Center of Atmospheric Environment and Equipment Technology (CICAEET), Nanjing University of Information Science and Technology, Nanjing 210044, China

Correspondence should be addressed to Beijing Chen; nbutimage@126.com

Received 7 April 2022; Accepted 21 June 2022; Published 9 July 2022

Academic Editor: Andrea Michienzi

Copyright © 2022 Beibei Zhang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In recent years, deep learning-based watermarking algorithms have received extensive attention. However, the existing algorithms mainly use the autoencoder to insert watermark automatically and ignore using the prior knowledge to guide the watermark embedding. In this paper, an end-to-end framework based on embedding guidance is proposed for robust image watermarking. It contains four modules, i.e., prior knowledge extractor, encoder, attacking simulator, and decoder. To guide the watermark embedding, the prior knowledge extractor providing chrominance and edge information of cover images is used to modify cover images before inserting the watermark by the encoder. To enhance the robustness of watermark extraction, the attacking simulator applying various differentiable attacks on the encoded images is introduced before extracting the watermark by the decoder. Experimental results show that the proposed algorithm achieves a good balance between invisibility and robustness and is superior to state-of-the-art algorithms.

1. Introduction

The unauthorized distribution of copies has become a threat to sharing of multimedia products. Hence, how to declare the ownership of the products is an urgent problem to be solved [1]. Digital watermarking technologies are widely used in copyright protection by embedding copyright information into digital products [2], such as digital literature, music, film, photography, and face portrait. Robustness against different attacks is significant for the practical application of digital watermarking. Traditionally, watermarking algorithms mainly rely on hand-crafted features to improve the robustness, such as applying various transforms [3–5] or using perceptual masking [6, 7]. The drawback to these hand-crafted algorithms is that they are not simultaneously robust to some types of distortions because different types of distortions often require different techniques [8]. Consequently, some deep learning-based algorithms

have been presented [9–23]. They usually utilize convolutional neural network (CNN) to design end-to-end architecture with an encoder and a decoder. In order to further improve robustness, some improvement measures are proposed. These improvements can be categorized into two classes, i.e., attacking simulation and model architecture design [10]. The summary of different watermarking algorithms is listed in Table 1.

1.1. Attacking Simulation. Zhu et al. [11] were the first to propose a robust watermarking network HiDDeN with an attacking simulator. The attacking simulator was inserted into the network to satisfy the end-to-end training. However, HiDDeN can only be robust to a single attack, such as JPEG, Gaussian blur, crop, and dropout. Then, Mellimi et al. [12] and Ahmadi et al. [13] improved the attacking simulator to resist combined attacks. Since JPEG compression attack is

TABLE 1: Review of different deep learning-based watermarking algorithms.

Improvements	References	Techniques	Attacks	Capacity
Attacking simulation	Zhu [11]	The first work to simulate attacks by inserting the noise layers	Crop, Gaussian, dropout, and JPEG	90 bits
	Mellimi [12]	Simulation of noise layers against agnostic attacks	JPEG, noise, and noise	1024 bits
	Ahmadi [13]	Simulation of noise layers to resist mixture attacks	Crop, Gaussian, resize, and JPEG	1024 bits
	Chen [14]	Simulation of differentiable JPEG quantization	JPEG	1024 bits
	Jia [15]	Combination of simulated and real JPEG in noise layer	JPEG, crop, and Gaussian	1024 bits
	Ying [16]	Training a network to simulate JPEG compression	JPEG, scaling, and Gaussian	A whole image
Model architecture design	Dhaya [17]	Lightweight CNN scheme U-net architecture	JPEG, Gaussian, and median	512 bits
	Fang [18]		Transparency, JPEG, and crop	128 bits
	Cun [19]	Combination of SplitNet and RefineNet	Crop and color	A whole image
	Mun [20]	Attention mechanism	JPEG, crop, filtering, and noise	512 bits
	Yu [21]	Generative adversarial network with attention mask	Noise, crop, and shift	A whole image
	Hao [22]	Generative adversarial network with a high-pass filter	Crop, Gaussian, and flip	30 bits
	Li [23]	Generative adversarial network with perceptual losses	Noise, filtering, and sharpen	1024 bits

nondifferentiable, some works [14–16] focused on JPEG compression simulation and improved the JPEG simulator by various differentiable methods to enhance the robustness against JPEG compression.

1.2. Model Architecture Design. Dhaya et al. [17] proposed a lightweight convolution neural network (LW-CNN) for the digital watermarking scheme, which had more resilience than other standard approaches. Fang et al. [18] exploited a template-based approach combined with U-Net to achieve better robustness. Cun et al. [19] used SplitNet and RefineNet to smooth watermarked regions for a better quality of watermarked images. Mun et al. [20] introduced attention mechanism into the watermarking field to achieve good performance in robustness against attacks. In addition, some notable algorithms with adversarial training [21–23] have greatly improved the perceptual quality of the watermarked images.

However, these existing CNN-based robust watermarking algorithms focus on attacking simulation and model architecture design before the watermark extraction. They do not consider prior knowledge to guide the watermark embedding. To further balance between invisibility and robustness, motivated by the traditional algorithms, some prior knowledge, such as the chrominance and edge saliency of cover images, is considered before the watermark embedding. The major contributions are as follows.

- (1) We propose a prior knowledge extractor to obtain the chrominance and edge saliency of cover images for guiding watermark embedding.
- (2) We propose an embedding guided end-to-end framework for robust watermarking based on the proposed prior knowledge extractor and attacking simulator.

- (3) We conduct a lot of empirical experiments to evaluate the performance of the proposed algorithm in terms of invisibility and robustness. Experimental results demonstrate that our algorithm achieves a good balance between invisibility and robustness and performs better than state-of-the-art algorithms.

2. Methods

In this section, the proposed framework is described in detail. The overall architecture and loss functions are presented in subsection 2.1. Then, each module is explained in subsections 2.2–2.6 one by one, i.e., prior knowledge extractor, encoder, attacking simulator, decoder, and discriminator.

2.1. Model Architecture. The main framework is presented in Figure 1. As shown in Figure 1, the proposed model is based on autoencoder structure, which consists of four modules: a prior knowledge extractor, an encoder, an attacking simulator, and a decoder. The prior knowledge extractor obtains prior knowledge to modify cover images for guiding watermark insertion. After that, the encoder hides the watermark into the modified cover image. Then, the attacking simulator performs various simulated attacks on encoded images as a network layer. Finally, the decoder extracts the watermark from attacked (or unattacked) encoded images. These modules achieve their objectives through the following loss functions.

The encoder aims to insert the watermark into the cover image invisibly. So, the distortion loss is used to limit the distortion of the encoded image by

$$L_D(I_{co}, I_{en}) = \|I_{co} - I_{en}\|_2^2, \quad (1)$$

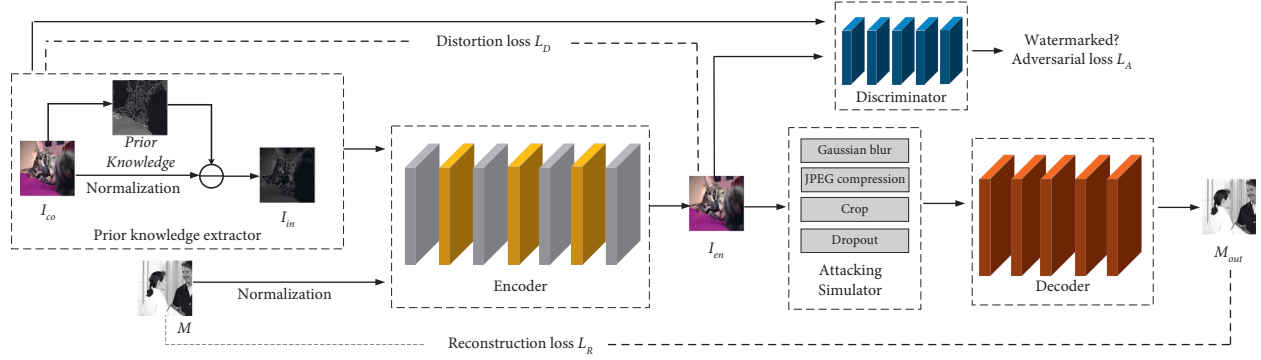


FIGURE 1: Overall architecture of the proposed model.

where I_{co} and I_{en} represent the cover image and encoded image, respectively.

The decoder wants to extract the watermark from the encoded images as much as possible. So, the reconstruction loss is adopted to improve the quality of the extracted watermark by

$$L_R(M, M_{out}) = \|M - M_{out}\|_2^2, \quad (2)$$

where M and M_{out} are the original watermark and extracted watermark, respectively.

The discriminator is used to judge whether the generated images are similar enough to the cover images. The discriminator and encoder compete with each other. So, the adversarial loss is considered to optimize the visual quality of the encoded image by

$$L_A(I_{co}, I_{en}) = \log(1 - D(I_{co})) + \log(D(I_{en})), \quad (3)$$

where D represents the discriminator.

Therefore, the total loss for the proposed framework is

$$L_{total} = \alpha L_D(I_{co}, I_{en}) + \beta L_R(M, M_{out}) + \gamma L_A(I_{co}, I_{en}), \quad (4)$$

where α , β , and γ are three hyper-parameters.

2.2. Prior Knowledge Extractor Module. Most existing deep learning-based algorithms mainly use the autoencoder to insert watermark automatically and ignore using the prior knowledge to guide the watermark embedding. According to the human visual system (HVS), people are less sensitive to modification in regions with rich chrominance and edge information [24–29]. So, the chrominance and edge saliency proposed in [30] are considered prior knowledge in this paper. The cover image is modified before watermark insertion to make the watermarking robust. Figure 2 depicts the flow diagram of our proposed prior knowledge extractor.

In order to obtain the chrominance information of the cover images, first, the cover image is converted into YC_bC_r color space by

$$\begin{aligned} Y &= 0.299R + 0.587G + 0.114B, \\ C_b &= 0.564(B - Y), \\ C_r &= 0.713(R - Y), \end{aligned} \quad (5)$$

where Y represents the luminance component and C_b and C_r represent chrominance components.

Then, the chrominance saliency $S_C(x)$ of a point x is obtained by

$$S_C(x) = 1 - \exp\left(-\frac{f_b^2(x) + f_c^2(x)}{\delta^2}\right), \quad (6)$$

where $f_b(x)$ and $f_c(x)$ are the normalization mappings of the C_b and C_r components, respectively, δ is a parameter set as 0.25 in this paper.

In order to obtain the edge information of cover images, the canny operator [31] is used to extract edge information. The edge saliency $S_E(x)$ of a point x is computed by

$$S_E(x) = \exp\left(-\frac{\text{Canny}(x) + 1}{\tau}\right), \quad (7)$$

where $\text{Canny}(x)$ represents the result calculated by the canny operator for a given point x and τ is a threshold set as 2 in this paper.

Finally, as is known to all, the stronger the chrominance and edge saliency are, the less sensitive the human eye is. So, the cover image is modified by

$$I_{in} = I_{co} - \left(1 - \frac{S_C(x) + S_E(x)}{2}\right), \quad (8)$$

where I_{co} is the original cover image after normalization and I_{in} is its modified one. According to (8), the greater the chrominance and edge saliency is, the smaller the modification of the cover pixel is, consequently, the relatively greater the change of cover pixel is in the watermark insertion.

2.3. Encoder Module. The architecture of the encoder network is illustrated in Figure 3. As shown in Figure 3, the encoder network has two parallel branches corresponding to the cover image and watermark image, respectively. One branch uses some convolutional layers to extract shallow detail features and deep semantic features of input normalized watermark images. The other branch uses a sequence of convolutional layers to extract features of the input cover image for merging with the features extracted from the watermark image. Specifically, in order to embed

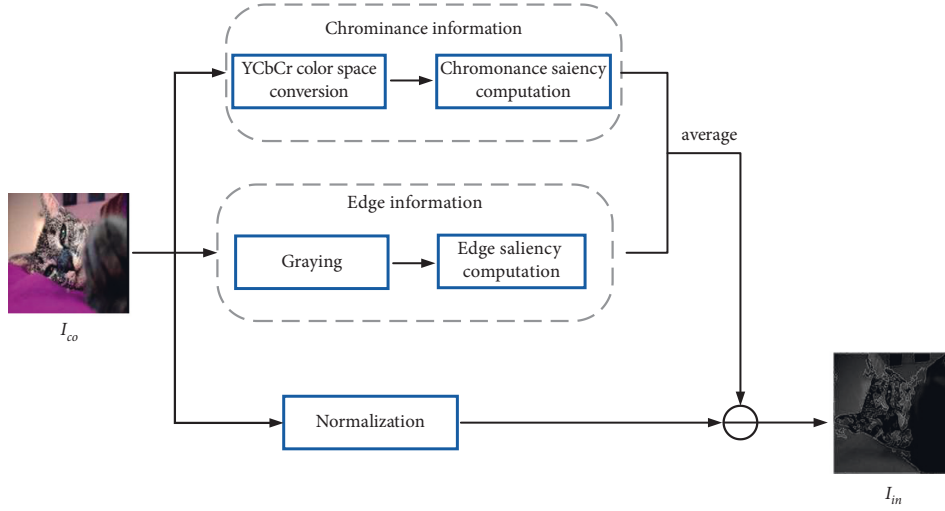


FIGURE 2: Flow diagram of the proposed prior knowledge extractor.

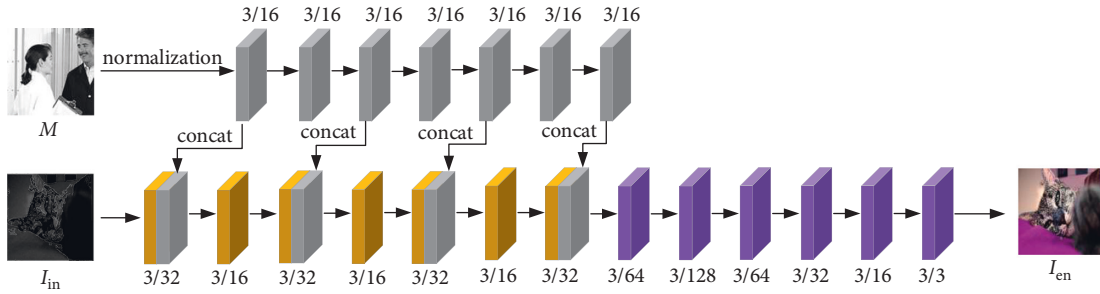


FIGURE 3: Architecture of the encoder. The numbers in the form m/n represent the kernel size (m) and the number of kernels (n) in each convolution layer.

watermark images into cover images, the encoder concatenates the features map extracted from each alternate layer of the watermark branch to the corresponding output features of the cover branch. Like [32], this concatenating process is repeated four times. Finally, the cover image and watermark are entirely fused as encoded images.

2.4. Attacking Simulator Module. In order to be robust against a variety of image distortions, as shown in Figure 1, an attacking simulator is inserted between the encoder and decoder to simulate various attacks by differentiable methods. Its parameters do not require to be updated during the entire network training process. Note that each iteration randomly selects one type of attack with equal probability. Specifically, [33], as shown in Figure 4, our attacking simulator includes four types of attacks: Gaussian blur, crop, JPEG compression, and dropout.

2.4.1. Gaussian Blur. Gaussian blur is also called Gaussian smoothing. It blurs the encoded images by performing a convolution operation with a Gaussian kernel. The larger the size of the convolution kernel, the stronger the blur attack.

2.4.2. Crop. Crop operation is simulated by randomly cropping out a small rectangle from the encoded images, namely, by replacing all the pixel values in this rectangle with zero. Specifically, the attack is simulated by multiplying with a 0–1 mask of the same size as the encoded image. In this mask, the region with pixel value 0 represents the cropped region, while the region with pixel value 1 represents the remaining region.

2.4.3. JPEG Compression. The steps of JPEG compression are composed of color space transformation, discrete cosine transform, quantization, and entropy coding. The sampling and discrete cosine transform steps are modeled by the max-pooling layer and convolution layer, respectively. Especially, as shown in Figure 5, the nondifferentiable quantization step is approximately simulated by performing JPEG-mask on the feature maps [11].

2.4.4. Dropout. Dropout attack is a common noise in image processing. It is implemented by arbitrarily replacing a certain ratio of pixels with zero. The detailed processing is similar to crop attack by multiplying with a 0–1 mask. The

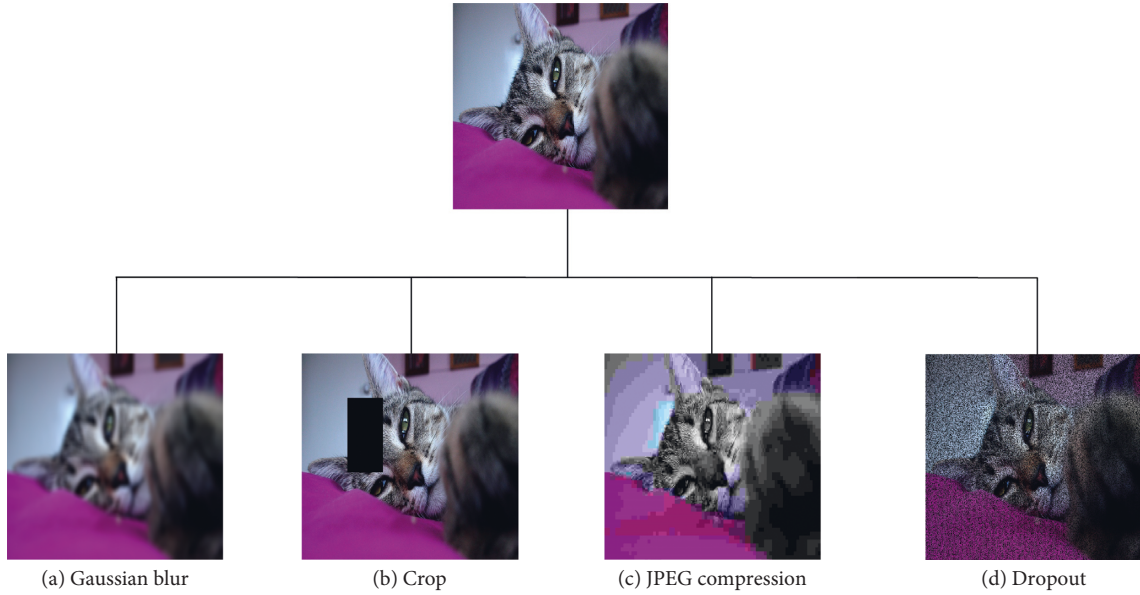


FIGURE 4: Samples of various attacks: (a)Gaussian blur; (b)crop; (c)JPEG compression; (d)dropout.

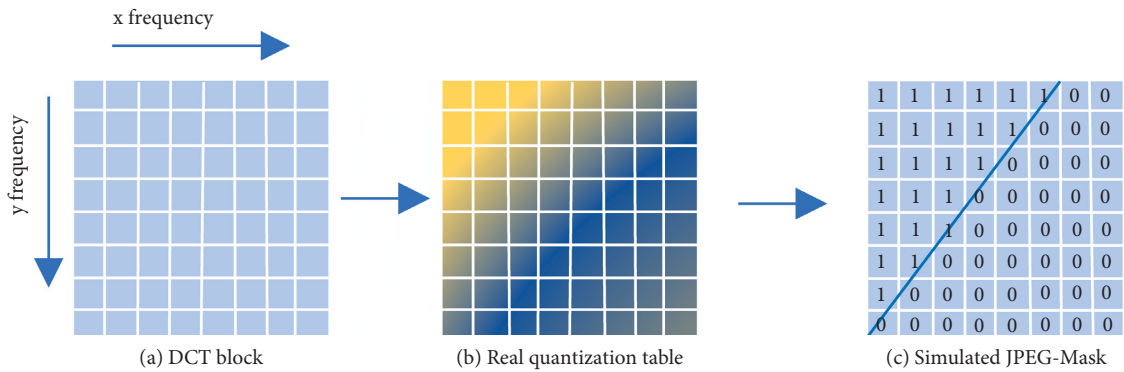


FIGURE 5: Simulation of nondifferentiable quantization step by JPEG-mask; (a)DCT block; (b)real quantization table; (c)simulated JPEG-mask.

difference is that the pixel values 0 and 1 are randomly distributed in the mask.

2.5. Decoder Module. In the end-to-end training, the decoder carries out the decoding procedure after encoding or attacking. The structure of the decoder is shown in Figure 6. The decoder takes the encoded or attacked image as input and extracts the watermark image. It uses seven Conv-BN-ReLU blocks to extract the watermark image from the input image. In this process, the function of convolutional operation is to extract features, and batch normalization (BN) speeds up the calculation while ReLU activation plays the filtering role. The final convolutional layer with a 3×3 kernel outputs watermark images.

2.6. Discriminator Module. The primary role of the discriminator is to improve the visual similarity between the encoded and cover images by adversarial training. The architecture of the discriminator is presented in Figure 7. It is similar to that of the decoder. The difference is that the

discriminator outputs binary classification results to judge whether the image contains the watermark or not. Therefore, the discriminator is built with five Conv-BN-ReLU blocks, an adaptive average pooling layer, a linear layer with a single output unit, and a Sigmoid activation layer.

3. Experimental Results and Analysis

In this section, experiments are carried out to prove the effectiveness and robustness of the proposed algorithm. The training datasets and experimental details are described in subsection 3.1. Then, the ablation experiments in subsection 3.2 are performed to demonstrate the improvements in the proposed algorithm. Finally, the robustness of the model for different types of attacks is tested in subsection 3.3.

3.1. Experimental Datasets, Implementation Details, and Evaluation Metrics

3.1.1. Experimental Datasets. 5,000 images randomly selected from the COCO dataset [34] are used as cover images.

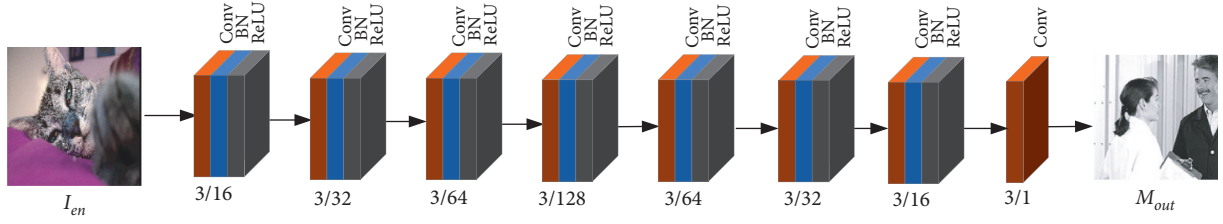


FIGURE 6: Architecture of the decoder. The numbers in the form m/n represent the kernel size (m) and the number of kernels (n) in each convolution layer.

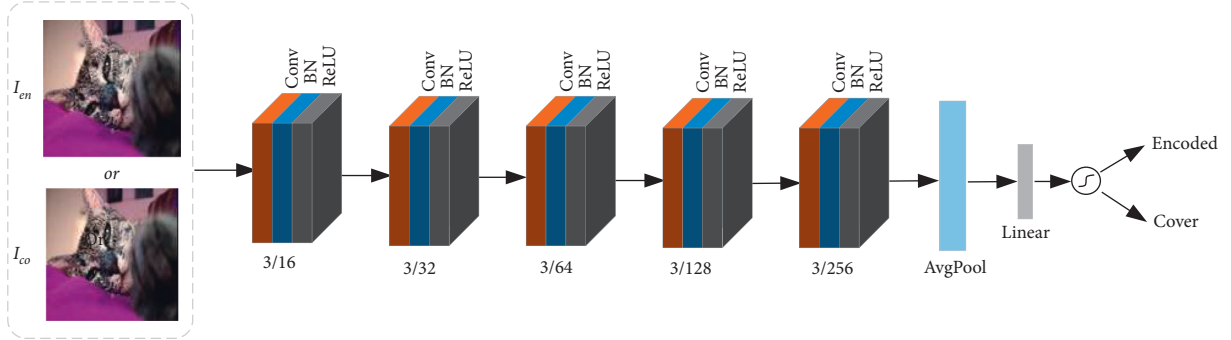


FIGURE 7: Architecture of the discriminator. The numbers in the form m/n represent the kernel size (m) and the number of kernels (n) in each convolution layer.

Three types of images are taken as watermark images for experiments. They are 5,000 logo images randomly selected from logo-2k+ [35], 5,000 digital number images from MNIST [36], and 5,000 general images from ImageNet [37]. These watermarks are converted into grayscale images before embedding. 5,000 cover images and each 5,000 watermark images are regarded as 5,000 pairs for the following experiments. Then, the cover images and watermark images are, respectively, divided into training/validation/testing sets according to the ratio of 8:1:1 and resized to 128×128 .

3.1.2. Implementation Details. The proposed watermarking model is trained iteratively using the ADAM optimizer [38] with an initial learning rate of $1.0e-3$. The batch size is set as 16. The weights in the loss function shown in (4) are set as $\alpha=0.3$, $\beta=0.7$, and $\gamma=0.001$. In addition, all simulated attacks have a hyperparameter governing the strengths: the kernel width ω of Gaussian blur is 3; quality factor QF of JPEG compression is 90; and ratios p of crop and dropout are 0.1 and 0.15, respectively.

3.1.3. Evaluation Metrics. The image visual quality is commonly evaluated by peak signal-to-noise ratio (PSNR) and structural similarity index metric (SSIM) metrics. Their definitions are given in the following.

Given two images U and V , the PSNR can be defined as

$$\text{PSNR}(U, V) = 10 \log_{10} \left(\frac{L^2}{\text{MSE}(U, V)} \right), \quad (9)$$

where L is the maximum pixel value, which is usually set as 255, MSE is mean squared error defined as

$$\text{MSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (U_i - V_i)^2}, \quad (10)$$

where n is the number of pixels.

The SSIM between two images U and V is defined as

$$\text{SSIM}(U, V) = \frac{(2\mu_U\mu_V + C_1)(2\sigma_{UV} + C_2)}{(\mu_U^2\mu_V^2 + C_1)(\sigma_U^2\sigma_V^2 + C_2)}, \quad (11)$$

where μ_U and μ_V are the means, σ_U and σ_V are the standard deviations, σ_{UV} is the cross-covariance of U and V , and C_1 and C_2 are two constants used to avoid a null denominator.

3.2. Ablation Experiments. Here, some ablation experiments are conducted to validate the proposed model. All the experiments are performed under the combined attacks with all four different types of attacks.

Firstly, we begin by analyzing what the prior knowledge extractor can do. Table 2 shows the average PSNR and SSIM values of 5,000 encoded images and 5,000 extracted watermarks with/without the extractor. As the results are shown, the visual quality of both the encoded images and extracted watermarks is improved after introducing the prior knowledge extractor. This is because the extractor obtains prior knowledge to find more suitable locations for watermark embedding.

Then, we verify the effectiveness of the attacking simulator. So, we compared the proposed models without and with the attacking simulator in the training stage. The results are shown in Table 2. As shown in Table 3, when the attacking simulator is considered, although the quality of the

TABLE 2: Performance comparison between the proposed model without and with the prior knowledge extractor.

Prior knowledge extractor	Encoded image		Extracted watermark	
	PSNR	SSIM	PSNR	SSIM
Without	37.56	0.9626	35.29	0.9595
With	40.69	0.9904	38.19	0.9722

TABLE 3: Performance comparison between the proposed model without and with an attacking simulator.

Attacking simulator	Encoded image		Extracted watermark	
	PSNR	SSIM	PSNR	SSIM
Without	41.02	0.9913	24.12	0.7824
With	40.69	0.9904	38.19	0.9722

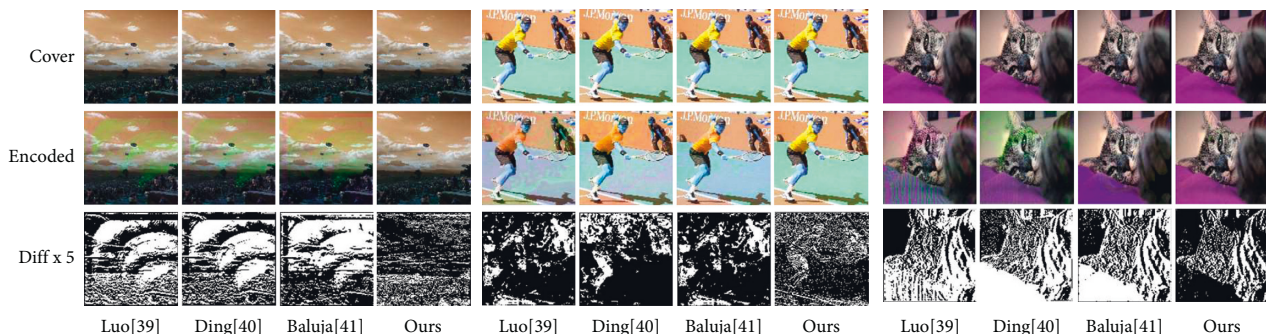


FIGURE 8: Some examples of the cover images (landscape, sportsman, and cat) and their corresponding encoded images, as well as their five times magnified differential images.

encoded images is sacrificed a little bit and the quality of extracted watermarks improves significantly. The PSNR values of the extracted watermarks increase from 24.12 dB to 38.19 dB and SSIM from 0.7824 to 0.9722.

The experimental results in Tables 2 and 3 show that either the prior knowledge extractor or attacking simulator is significant to the robust watermarking.

3.3. Comparison Experiments with Other Algorithms. In order to further evaluate the performance of the proposed algorithm, our algorithm is compared with some existing deep learning-based algorithms [39–41] in terms of both invisibility and robustness.

3.3.1. Invisibility. The challenge for digital watermarking is to improve robustness while keeping invisibility. Figure 8 shows the visual comparison of different watermarking algorithms. In addition, Table 4 presents their corresponding numerical results by PSNR and SSIM. It can be observed from Figure 8 and Table 4 that the watermarks are invisible in the encoded images for the proposed algorithm with high PSNR and SSIM values, while it is not the case for the other three algorithms who suffer from a little color bias. This is due to the use of prior knowledge for guiding watermark insertion in our algorithm.

TABLE 4: PSNR and SSIM values of three encoded images in Figure 8 for different algorithms.

Algorithms	Landscape		Sportsman		Cat	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Luo [39]	30.19	0.883	31.34	0.896	30.46	0.886
Ding [40]	30.53	0.892	32.78	0.912	31.89	0.905
Baluja [41]	31.42	0.9012	33.97	0.924	32.75	0.917
Ours	36.71	0.957	38.12	0.963	36.92	0.959

3.3.2. Robustness. In order to test the robustness, the encoded images are carried out in five different types of attacks. Table 5 presents the average PSNR and SSIM values of 5,000 encoded images and 5,000 watermark images for four compared algorithms. In addition, Figure 9 shows some visual samples of the extracted watermarks. It can be observed from Table 5 and Figure 9 that the proposed algorithm achieves the best performance for all five types of attacks in both numerical and visual aspects, especially for the combined attack. Although the encoded images are distorted under various attacks, our algorithm can preserve watermark fidelity to a great extent with few errors. However, it is not the case for the other three algorithms, whose extracted watermarks suffer from some errors with some noise in vision. This is attributable to the watermarking guidance of prior knowledge and the consideration of

TABLE 5: Comparison of robustness against different types of attacks.

Attacks	Algorithms	Encoded image				Extracted watermark			
				Logo		MNIST		ImageNet	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Gaussian blur ($\omega = 9$)	Luo [39]	33.79	0.9213	31.49	0.9109	32.76	0.9267	30.38	0.9015
	Ding [40]	32.56	0.9146	30.24	0.8937	31.95	0.9086	28.89	0.8812
	Baluja [41]	34.24	0.9322	32.62	0.9148	33.97	0.9275	31.04	0.9068
	Ours	41.11	0.9751	36.24	0.9641	37.87	0.9753	34.92	0.9549
JPEG compression ($QF = 60$)	Luo [39]	33.08	0.9225	30.37	0.9021	31.43	0.9078	29.12	0.8994
	Ding [40]	32.67	0.9114	29.73	0.8864	30.82	0.9026	28.23	0.8801
	Baluja [41]	34.49	0.9511	31.39	0.9115	32.74	0.9248	30.59	0.9082
	Ours	38.35	0.9657	34.17	0.9512	35.86	0.9573	33.11	0.9448
Crop ($p = 0.4$)	Luo [39]	32.93	0.9325	31.37	0.9121	32.66	0.9264	30.25	0.9027
	Ding [40]	32.20	0.9007	30.88	0.8964	32.05	0.9128	30.08	0.8901
	Baluja [41]	34.36	0.9461	32.59	0.9323	33.74	0.9456	31.23	0.9119
	Ours	39.45	0.9727	37.06	0.9605	38.85	0.9773	36.39	0.9597
Dropout ($p = 0.45$)	Luo [39]	34.16	0.9321	32.45	0.9409	33.76	0.9487	31.07	0.9339
	Ding [40]	32.64	0.9145	30.36	0.9047	31.83	0.9102	29.99	0.8995
	Baluja [41]	33.97	0.9343	32.82	0.9222	33.75	0.9298	32.01	0.9186
	Ours	39.18	0.9710	35.64	0.9586	36.83	0.9642	34.75	0.9503
Combined	Luo [39]	32.17	0.9189	30.01	0.8832	30.95	0.9045	28.67	0.8974
	Ding [40]	31.01	0.9013	28.47	0.8813	31.08	0.8942	27.84	0.8757
	Baluja [41]	33.12	0.9387	30.96	0.9102	31.89	0.9211	30.22	0.9032
	Ours	37.64	0.9724	34.11	0.9505	35.79	0.9568	34.68	0.9501

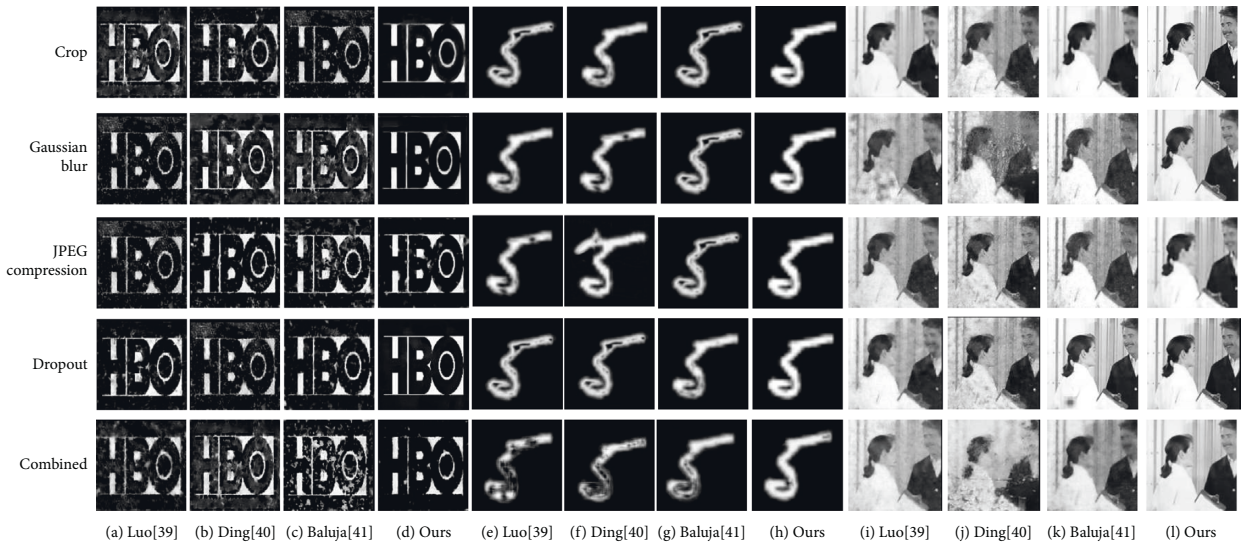


FIGURE 9: Visual comparison of robustness against different kinds of attacks. Samples (a)–(d) are from Logo-2k, (e)–(h) from MNIST, and (i)–(l) from ImageNet.

attacking simulator in our algorithm. Regarding three different types of watermark images, all the compared algorithms perform best on the MNIST watermarks. The main reason is that the other two types of watermark images are more complex and contain more semantic information, which results in more difficulty in the watermark extraction.

In addition, we evaluate the generalization performance of different watermarking algorithms against attacks different from those in the training stage in two aspects, i.e., different attack levels and different attack types.

3.3.3. Different Attack Levels. Figure 10 shows the average PSNR values of the extracted watermark images under different attack levels. As shown in Figure 10, our algorithm still performs better than the other three algorithms when being attacked by different levels of various attacks. In addition, the performance of all four compared algorithms decreases with the increase in attack levels.

Different attack types. To evaluate the performance in resisting the attacks that were not considered during the training stage, we select four kinds of black-box image

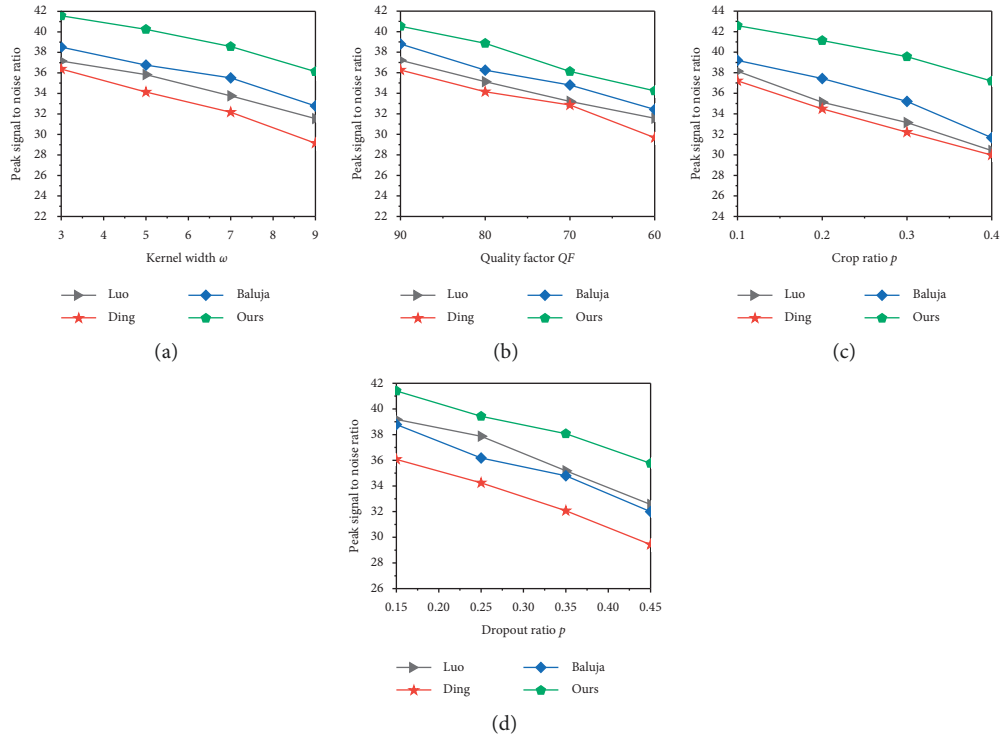


FIGURE 10: Comparison with other algorithms under different attacks with different levels: (a)Gaussian blur; (b)JPEG compression; (c)crop; (d)dropout.

TABLE 6: Comparison with other algorithms against the attack types different from those in the training stage.

Algorithms	Resizing ($T=2$)	Medium blur ($\omega=3$)	Salt and pepper noise ($p=0.2$)	Gaussian noise ($\sigma=1.0$)
Luo [37]	27.97	29.48	30.15	29.75
Ding [38]	26.64	28.57	29.31	28.43
Baluja [39]	28.89	30.32	31.67	30.46
Ours	32.87	33.91	34.02	33.52

attacks (resizing, medium blur, salt and pepper noise, and Gaussian noise) to test the model. The levels of these attacks are as follows: the scaling factor T of resizing is 2; kernel width ω of medium blur is 3; ratio p of salt and pepper noise is 0.2; and standard deviation σ of Gaussian noise is 1.0. Table 6 shows the average PSNR values of the extracted watermark images of different algorithms. As can be seen from Tables 5 and 6, the proposed algorithm still maintains higher PSNR values than the other three algorithms, though its performance decreases when facing attacks different from the training stage.

4. Conclusion

In this paper, we propose an embedding guided end-to-end framework for robust image watermarking. In this algorithm, a prior knowledge extractor and attacking simulator are introduced to guide watermarking embedding and enhance the robustness of watermark extraction, respectively. The experiment results demonstrate that, compared to the existing algorithms, the proposed algorithm performs better in both invisibility and robustness. However, the proposed

algorithm does not consider other common attacks in practical application, such as printing, screen photography, and geometric transformation. Therefore, in the future, we will focus on the simulation of these attacks and study the deep learning-based watermarking algorithms against these attacks.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the Open Fund of Advanced Cryptography and the System Security Key Laboratory of Sichuan Province (Grant no. SKLACSS-202113) and the

National Natural Science Foundation of China (Grant no. 62072251), the PAPD fund.

References

- [1] H. Wu, G. Liu, Y. Yao, and X. Zhang, "Watermarking neural networks with watermarked images," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 7, pp. 2591–2601, 2020.
- [2] U. H. Panchal and R. Srivastava, "A comprehensive survey on digital image watermarking techniques," in *Proceedings of the 2015 Fifth International Conference on Communication Systems and Network Technologies*, pp. 591–595, Gwalior, India, April 2015.
- [3] H. Nazir, I. S. Bajwa, M. Samiullah, A. Waheed, and M. Muhammad, "Robust secure color image watermarking using 4D hyperchaotic system, DWT, HbD, and SVD based on improved FOA algorithm," *Security and Communication Networks*, vol. 2021, Article ID 6617944, 17 pages, 2021.
- [4] H. C. Fu, Z. F. Zhao, and X. L. He, "Improving Anti-compression Robustness of JPEG Adaptive Steganography Based on Robustness Measurement and DCT Block Selection," *Security and Communication Networks*, vol. 2021, Article ID 9153468, 15 pages, 2021.
- [5] C. S. Yang, J. B. Li, U. A. Bhatti, J. Liu, J. Ma, and M. Huang, "Robust zero watermarking algorithm for medical images based on zernike-DCT," *Security and Communication Networks*, vol. 2021, Article ID 4944797, 8 pages, 2021.
- [6] W. F. Qi, Y. X. Liu, S. R. Guo, X. Wang, and Z. Guo, "An adaptive visible watermark embedding method based on region selection," *Security and Communication Networks*, vol. 2021, Article ID 6693343, 11 pages, 2021.
- [7] K. Zebbiche, F. Khelifi, and K. Loukhaoukha, "Robust additive watermarking in the DTCWT domain based on perceptual masking," *Multimedia Tools and Applications*, vol. 77, no. 16, Article ID 21304, 2018.
- [8] M. Asikuzzaman and M. R. Pickering, "An overview of digital watermarking," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 9, pp. 2131–2153, 2017.
- [9] Y. Quan, H. Teng, Y. Chen, and H. Ji, "Watermarking deep neural networks in image processing," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 5, pp. 1852–1865, 2021.
- [10] H. Kandi, D. Mishra, and S. R. S. Gorthi, "Exploring the learning capabilities of convolutional neural networks for robust image watermarking," *Computers & Security*, vol. 65, pp. 247–268, 2017.
- [11] J. Zhu, R. Kaplan, J. Johnson, and L. Fei, "Hidden: hiding data with deep networks," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 657–672, Munich, Germany, September 2018.
- [12] S. Mellimi, V. Rajput, I. A. Ansari, and C. W. Ahn, "A fast and efficient image watermarking scheme based on deep neural network," *Pattern Recognition Letters*, vol. 151, pp. 222–228, 2021.
- [13] M. Ahmadi, A. Norouzi, N. Karimi, E. Ali, and S. Samavi, "ReDMark: framework for residual diffusion watermarking based on deep networks," *Expert Systems with Applications*, vol. 146, Article ID 113157, 2020.
- [14] B. J. Chen, Y. Q. Wu, G. Coatrieux, X. Chen, and Y. Zheng, "JSNet: a simulation network of JPEG lossy compression and restoration for robust image watermarking against JPEG attack," *Computer Vision and Image Understanding*, vol. 197–198, Article ID 103015, 2020.
- [15] Z. Y. Jia, H. Fang, and W. M. Zhang, "MBRS: enhancing robustness of DNN-based watermarking by mini-batch of real and simulated JPEG compression," in *Proceedings of the 29th ACM International Conference on Multimedia*, pp. 41–49, Chengdu, China, October 2021.
- [16] Q. Ying, H. Zhou, X. Zeng, H. Xu, Z. Qian, and X. Zhang, "Hiding Images into Images with Real-World Robustness," 2021, <https://arxiv.org/abs/2110.05689>.
- [17] R. R., "Light weight CNN based robust image watermarking scheme for security," *Journal of Information Technology and Digital World*, vol. 3, no. 2, pp. 118–132, 2021.
- [18] H. Fang, D. Chen, Q. Huang et al., "Deep template-based watermarking," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 4, pp. 1436–1451, 2020.
- [19] X. Cun and C. M. Pun, "Split Then Refine: Stacked Attention-Guided ResUNets for Blind Single Image Visible Watermark Removal," 2020, <https://arxiv.org/abs/2012.07007>.
- [20] S. M. Mun, S. H. Nam, H. Jang, D. Kim, and H.-K. Lee, "Finding robust domain from attacks: a learning framework for blind watermarking," *Neurocomputing*, vol. 337, pp. 191–202, 2019.
- [21] C. Yu, "Attention based data hiding with generative adversarial networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 1120–1128, NY, USA, February 2020.
- [22] K. Hao, G. Feng, and X. P. Zhang, "Robust image watermarking based on generative adversarial network," *China Communications*, vol. 17, no. 11, pp. 131–140, 2020.
- [23] Q. Li, X. Y. Wang, B. Ma et al., "Concealed attack for robust watermarking based on generative model and perceptual loss," *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.
- [24] B. J. Chen, X. Liu, Y. H. Zheng, G. Y. Zhao, and Y. -Q. Shi, "A robust GAN-generated face detection method based on dual-color spaces and an improved xception," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 6, pp. 3527–3538, 2022.
- [25] H. Y. Qi, D. Zhao, and J. Y. Zhao, "Human visual system based adaptive digital image watermarking," *Signal Processing*, vol. 88, no. 1, pp. 174–188, 2008.
- [26] T. Sridevi and S. Fathima, "Watermarking algorithm based using genetic algorithm and HVS," *International Journal of Computer Application*, vol. 74, no. 13, 2013.
- [27] F. Kabir and M. N. Kabir, "A Block-based RDWT-SVD Image Watermarking Method Using Human Visual System Characteristics," *The Visual Computer*, vol. 36, no. 1, pp. 19–37, 2020.
- [28] B. J. Chen, W. J. Tan, C. Coatrieux, Y. H. Zheng, and Y. -Q. Shi, "A serial image copy-move forgery localization scheme with source/target distinguishment," *IEEE Transactions on Multimedia*, vol. 23, pp. 3506–3517, 2021.
- [29] Y. L. Bei, S. Qiao, M. X. Liu, Q. Zhang, and X. R. Zhu, "A color image watermarking scheme Against geometric rotation attacks based on HVS and DCT-DWT," in *Proceedings of the 2018 International Conference on Security, Pattern Analysis, and Cybernetics (SPAC)*, pp. 343–347, Jinan, China, December 2018.
- [30] L. Zhang, Z. Gu, and H. Li, "SDSP: a novel saliency detection method by combining simple priors," in *Proceedings of the 2013 IEEE International Conference on Image Processing*, pp. 171–175, Melbourne, VIC, Australia, September 2013.
- [31] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679–698, 1986.

- [32] B. J. Chen, J. X. Wang, Y. Y. Chen, Z. Jin, H. J. Shim, and Y. Q. Shi, "High-capacity robust image steganography via adversarial network," *KSI Transactions on Internet and Information Systems (TIIS)*, vol. 14, no. 1, pp. 366–381, 2020.
- [33] M. Jamali, N. Karim, P. Khadivi, Z. Jin, H. J. Shim, and Y. Q. Shi, "Robust Watermarking Using Diffusion of Logo into Autoencoder Feature Maps," 2021, <https://arxiv.org/abs/2105.11095>.
- [34] T. Y. Lin, M. Maire, S. Belongie et al., "Microsoft coco: common objects in context," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 740–755, Zurich, Switzerland, September 2014.
- [35] J. Wang, W. Min, S. Hou et al., "Logo-2K+: a large-scale logo dataset for scalable logo classification," in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 6194–6201, NY, USA, February 2020.
- [36] Y. L. Cun, "The MNIST Database of Handwritten Digits," 1998.
- [37] J. Deng, W. Dong, R. Socher, F. F. Li, L. J. Li, and K. Li, "ImageNet: a large-scale hierarchical image database," in *Proceedings of the 2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Miami, FL, USA, June 2009.
- [38] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," 2014, <https://arxiv.org/abs/1412.6980>.
- [39] X. Luo, R. Zhan, H. Chang, F. Yang, and M. Peyman, "Distortion agnostic deep watermarking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Article ID 13557, Seattle, WA, USA, June 2020.
- [40] W. P. Ding, Y. R. Ming, Z. H. Cao, and C. T. Lin, "A generalized deep neural network approach for digital watermarking analysis," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 6, no. 3, pp. 1–15, 2021.
- [41] S. Baluja, "Hiding images within images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 7, pp. 1685–1697, 2020.

Research Article

Semantic Modeling and Pixel Discrimination for Image Manipulation Detection

Ziyu Xue ^{1,2}, Xiuhua Jiang,^{1,3} and Qingtong Liu ²

¹School of Information and Communication Engineering, Communication University of China, Beijing, China

²Academy of Broadcasting Science, NRTA, Beijing, China

³Peng Cheng Laboratory, Shenzhen, China

Correspondence should be addressed to Ziyu Xue; xzy_88@126.com

Received 27 January 2022; Revised 4 March 2022; Accepted 12 April 2022; Published 5 May 2022

Academic Editor: Beijing Chen

Copyright © 2022 Ziyu Xue et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Image manipulation methods, such as the copy-move, splicing, and removal methods, have become increasingly mature and changed the common perception of “seeing is believing.” The credibility of digital media has been seriously damaged with the development of image manipulation methods. Most image manipulation detection methods detect traces of tampering pixel by pixel. As a result, the detected manipulation areas are separated, which results in insufficient consideration of content manipulation at the object level. In this paper, the detection of image manipulation areas based on forgery object detection and pixel discrimination is proposed. Specifically, the pixel-level detection branch resamples features and uses an LSTM to detect manipulations, such as resampling, rotation, and cropping. The goal of the forgery object detection branch, which is based on Faster R-CNN, is to extract the regions of interest and analyze the regions with high contrast as well as the forgery objects of the image. Furthermore, the fused heatmaps of the two branches are integrated with the object detection results. The noise in the heatmaps is shielded based on the forgery object information of the region proposal network. Experimental results on multiple standard forgery datasets have demonstrated the superiority of our proposed method compared with the state-of-the-art methods.

1. Introduction

With the rapid evolution of digital image manipulation, digital images can be tampered with or forged through various methods, i.e., the splicing, copy-move, and image removal methods. When splicing is implemented, a portion of the source image splits into the target image to form a new image, as shown in Figure 1(a). The copy-move method is used to paste an area of an image into the same image, as shown in Figure 1(b). In the removal method, an area in the image is removed and the area restored, as shown in Figure 1(c).

Early image manipulation detection methods employed deep neural networks to determine the type of manipulation in advance. A detection method can only solve one kind of manipulation problem. This kind of method, which is referred to as known manipulation-based detection, uses the

Daubechies wavelet features to detect image patches [1] and edge reinforcement methods to build a multitask detection task [2]. Chen et al. [3] propose a parallel deep neural network scheme BusterNet for image copy-move forgery localization. With the development of manipulation methods, it has become easy to superimpose multiple image manipulations. Therefore, it is increasingly challenging to detect manipulated images based on unknown manipulation types.

The first line is splicing, the second line is copy-move, and the third line is removal.

Unknown manipulation-based detection has more significant applications than the above approach. Most existing detection approaches [4, 5] combine resampling and deep learning features to detect manipulated regions. Park et al. [6] utilized double JPEG compression features, which were merged with the additional information in the quantization

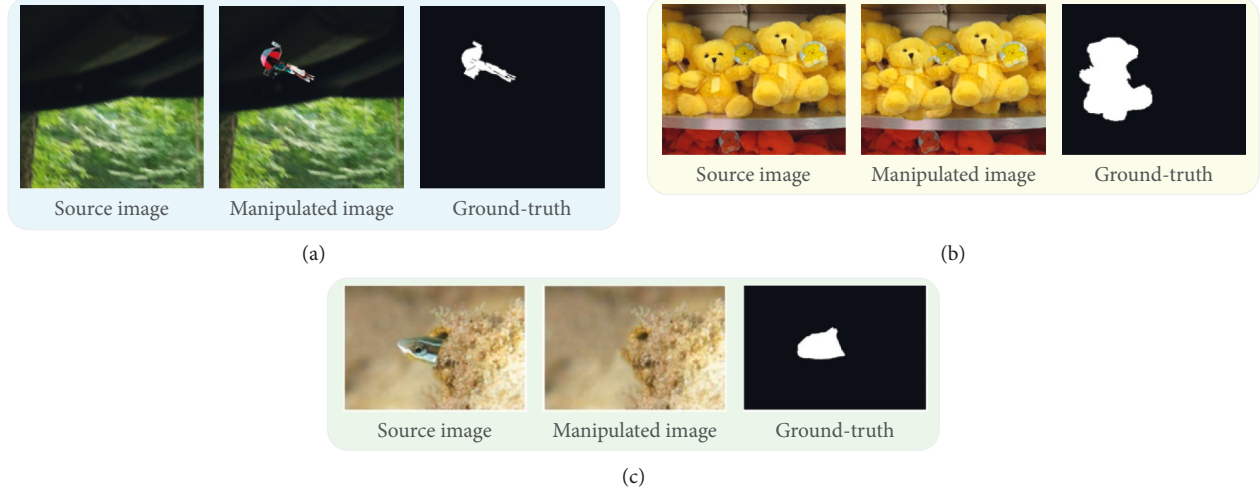


FIGURE 1: Image manipulation detection method. (a) Splicing. (b) Copy-move. (c) Removal.

table, to determine whether the image contains manipulated areas and locate them. Zhou et al. [7] proposed a framework that effectively combined a region proposal network (RPN) to localize the synthesis region at the object level and carry out a two-stream fine-grained bilinear pooling operation. Xiao et al. [8] proposed an approach that combines a coarse convolutional neural network (C-CNN) and a refined CNN (R-CNN) to learn the differences in the image properties to catch manipulated images. Chen et al. [9] utilize the dual-color spaces and improve the Xception architecture to detect GAN-generated faces. However, most of these approaches are pixel-level oriented. We find that image manipulation is more likely to occur at the object level. The image manipulation regions can be detected at the pixel level and object level to simultaneously contribute to improving the detection accuracy.

We have developed an approach, whose architecture is shown in Figure 2, which combines both pixel-level and object-level information. Our method consists of four parts, i.e., a forgery object-level branch, a pixel-level detection branch, a feature fusion module, and an integrated fusion module. First, the forgery object-level branch is used to extract the image features using CNNs and feed the features into the feature fusion module. Then, the ROIs are obtained based on Faster R-CNN [10]. Inspired by Bappy et al. [5], the pixel-level detection branch is designed to divide the image into 8×8 patches and resample them one by one.

Meanwhile, the LSTM learns to establish the temporal correlations between patches. Equipped with the outputs of two branches combined, the feature fusion module uses a decoder to reconstruct the features, and the heatmaps of the manipulation area are generated. The manipulation areas from the pixel-level branch are used to fine-tune the forgery object-level branch to achieve accurate forgery area labeling. The contributions of our work are summarized as follows:

- (1) We propose a novel two-branch image manipulation detection framework consisting of a forgery object-level branch and a pixel-level branch. The image manipulation region is refined by two fusion

modules, making our work significantly different from other state-of-the-art methods.

- (2) We employ the region of interest in the forgery object-level branch to optimize the heatmap in the feature fusion module. The noise in the heatmap is masked by bounding boxes to decrease the error caused by the pixel-level detection branch and improve the detection precision of the manipulated image regions.
- (3) Extensive experiments on four benchmarks have demonstrated the effectiveness of our proposed method.

The rest of this paper is organized as follows. In Section 2, we summarize the current image manipulation detection technologies. In Section 3, we elaborate on the details of our proposed method. The conducted experiments and analysis are presented in Section 4. Finally, in Section 5, we conclude the paper.

2. Related Work

Researches on detecting unknown manipulation type images consist of various approaches. Some earlier approaches [4, 6, 11] are based on manually designed features. Wu et al. [11] propose a method to divide the image into blocks and extract the resampling features for each block. A deep neural network is utilized to construct a classifier and a Gaussian conditional random field model to create a thermodynamic diagram. Meanwhile, they use the random walk method to locate the synthetic region.

Some approaches that are based on adaptive feature extraction are proposed to reduce the limitations of manual design features and improve the method's adaptability. Bappy et al. [5] construct a two-branch manipulation image detection architecture by combining the resampling feature, LSTM, and encoder-decoder architecture. A resampling detection model is utilized to extract the resampling feature of the image from each patch, and the LSTM establishes the

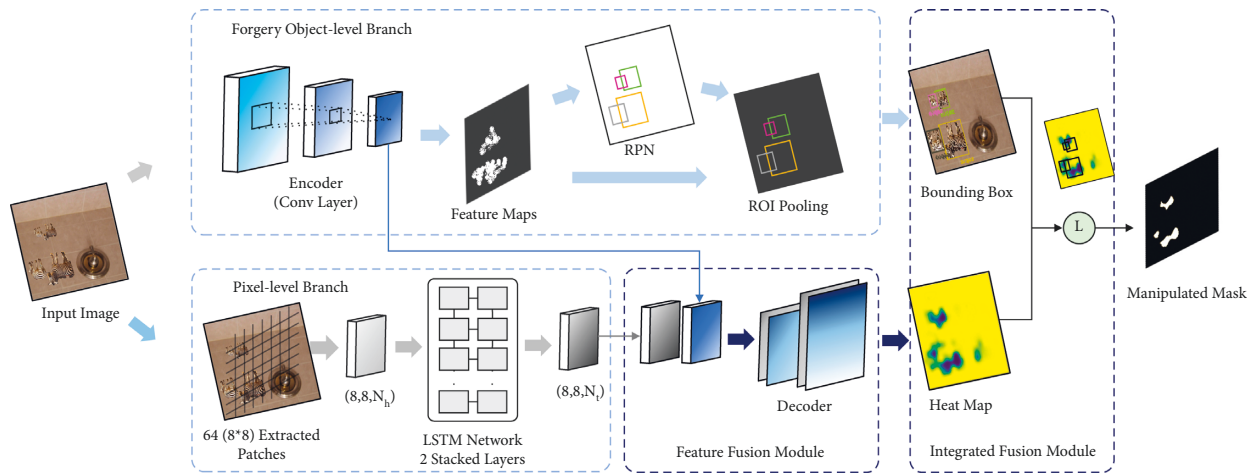


FIGURE 2: The overall architecture of our method. Forgery object-level branch: the features are extracted from the whole image using several convolutional layers. The forgery object information is extracted using encoder to form the feature maps to RPN. Pixel-level branch: the image is divided into 8×8 image patches, and the resampling features are extracted for each patch, combined with LSTM to build a temporal relationship. Feature fusion module: the forgery object-level features in the upper branch and the pixel-level features in the lower branch are integrated to produce the heat maps using a decoder. Integrated fusion module: according to the forgery object information of the RPN network, the noises in the heat maps are masked.

correlation between patches. The encoder is used to capture the spatial information of the image. After fusing the frequency and spatial features, the decoder is used to enlarge the feature to obtain the synthetic region located at the pixel level.

Mohammed et al. [12] use the CNN to obtain the abnormal image boundary features and the LSTM to establish the connection between image patches. A separate detection model is used to obtain more accurate detection results to enhance the detection effect of the copy-move manipulation image. Mazumdar et al. [13] present a two-stream encoder-decoder network. The first stream extracts the noise residuals to learn the low-level features through the encoder of the high-pass filter. The second stream extracts high-level features from the RGB values of the input image. The feature maps of both streams generated pixel-level predictions. Some other typical methods also detect pixel modifications such as resampling, copy-move, and removal.

By summarizing the above work, forgers mainly perform manipulation on objects, such as a car and sofa. Therefore, the forgery detection is based on object-level to explore the semantic information. Meanwhile, pixel-level detection is more accurate, especially at detecting edges of fake ones. It is necessary to combine both object- and pixel-level information to detect forgeries. Based on this, we present a novel framework to effectively detect the manipulation region, in which both object- and pixel-level features contribute to the detection results.

3. Network Architecture Overview

The purpose of the proposed framework is to detect image manipulations. A multitask framework is employed to model both object-level and pixel-level structures and consists of a forgery object-level branch and a pixel-level branch. A feature fusion module is further used to fuse both

forgery object-level and pixel-level features. The generated heatmaps are merged with the forgery object information of the manipulated areas through an integrated fusion module to detect the manipulated areas.

3.1. Forgery Object-Level Detection Branch. We utilize Faster R-CNN in the forgery object-level branch to detect manipulation areas. A convolutional network is used to learn manipulation features, and the RPN is utilized to generate ROIs for bounding box regression.

3.1.1. Feature Extraction Network. It is essential in image manipulation detection to extract features using convolutional neural networks and ensure that the classifier can learn to discriminate the manipulated areas. In our work, we employ ResNet-101 to extract image features. Specifically, we utilize different convolution kernels to locate the manipulated area. In the first layer, the image is taken as input with dimensions of $256 \times 256 \times 3$. The network contains multiple convolutions, pooling layers, and activation functions. The residual module utilizes a parameter-free shortcut connection to optimize the residual mapping and model training.

Following [5], we utilize a convolution kernel size of 3×3 to generate 32, 64, 128, and 256 feature maps. Each residual unit in the network generates a set of feature maps normalized by batch processing in the convolutional layer. The rectified linear unit (ReLU) function is utilized as an activation function followed by a max-pooling layer with a stride of two at the end of each residual unit.

3.1.2. Region Proposal Network. The RPN in Faster R-CNN is used to extract the proposed regions. Compared with the selective search method, RPN is more efficient and easier to combine with Faster R-CNN. As the anchor is the center

point of the sliding window in the original pixel space, we use the anchor as the center point to generate the proposed regions.

We first map the ROIs to the corresponding area on the feature map and shape the different ROIs to a fixed size. We take the maximum pixel value of each divided region. Then, each ROI will have a fixed size.

The loss of the forgery object-level branch is composed of two items, i.e., classification loss and regression loss, which are based on the RPN in Faster R-CNN. The i -th anchor can be represented as

$$L(\{p_i\}, \{t_i\}) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*), \quad (1)$$

where p is denoted as the probability of a manipulation area, p^* represents the ground-truth label, t is the 4-dimensional bounding box, N_{cls} and N_{reg} represent the RPN network batch and the number of anchors at each location, respectively, L_{cls} is the standard cross-entropy loss for the RPN network, L_{reg} is the smooth L_1 regression loss for the proposal bounding boxes, and the hyperparameter λ is utilized to balance the two loss items with a value of 10. The output of the forgery object-level branch is Z_{Object} , as shown in Figure 3.

3.2. Pixel-Level Detection Branch. The pixel-level detection branch is used to detect the natural statistics of copy-move, splicing, and removal, consisting of a resampling feature extraction network and an LSTM network. Following [5], we utilize the pixel-level features as the input of the feature fusion module.

3.2.1. Resampling Feature Extraction Network. Mahdian et al. [14] proposed a resampling detection approach using the Radon transform. They employed the Laplacian filter to resample each patch after the Radon transform. Bappy et al. [5] utilized the Radon transform to detect manipulations such as copy-move, splicing, and removal. They detected the tampered regions by distorting the natural statistics at the boundary. The Radon transform was proven to be effective in distinguishing manipulated and nonmanipulated patches.

Following [5], we set the size of the input image to $256 \times 256 \times 3$ and extracted 64 nonoverlapping patches from the images. The size of each block is $32 \times 32 \times 3$. Then, we utilize the square root of the 3×3 Laplacian filter to generate a linear prediction error for each patch. To prevent the periodic correlation of resampling features in linear prediction loss, the Radon transform is used to accelerate the gradient descent along with ten angles of projection. Ultimately, the fast Fourier transform (FFT) is applied to obtain the periodic signal. We balance the size of the patch and resize it to 32×32 to capture the resampling features of additional information.

3.2.2. LSTM Networks. The LSTM networks are utilized to establish the relationship between patches to analyze

manipulations in the overall image. Following [5], we use a Hilbert curve to convert the multidimensional problem into a single dimension to capture the correlation between patches and guarantee the local spatial positioning for patches.

From Figure 4(a), we find the results of the Hilbert curve on the image. All the patches are connected in order. The Hilbert curve includes ‘‘cups’’ and ‘‘joins.’’ A square with one open side represents a ‘‘cup.’’ The vector connections of two ‘‘cups’’ are called ‘‘joins.’’ Every cup has an entry point and an exit point. In Figure 4(b), a cup is marked with a dashed box from Figure 4(a). The curve starts at the entry point (red) and ends at the exit point (green). Meanwhile, the curve traverses four adjacent squares connected to the next cup through a dotted line. As a result, the order of the input that is fed into the LSTM network is established.

The LSTM network takes the patches associated with the Hilbert curve one by one as input and learns the relationship between adjacent patches by calculating the logarithmic interval. In our work, we employ 64 steps in the LSTM network, where each step represents a patch, and a 64-dimensional feature vector is obtained in the last layer of the LSTM. First, we denote the n -th feature of the LSTM as F_n ($F_n \in \mathbb{R}^{1 \times N_h}$) and the feature map as N_t . The next feature from the LSTM network can be represented as

$$F_n' = F_n \cdot W_n + B_n, \quad (2)$$

where $W_n \in \mathbb{R}^{N_h \times N_t}$ is a matrix and $B_n \in \mathbb{R}^{1 \times N_t}$ is the bias.

Following [5], we choose $N_h = 128$ and $N_t = 64$ in our experiment, and each patch can obtain the feature matrix of $64 \times N_t$ and is reshaped to $8 \times 8 \times N_t$. We set the cross-entropy loss as

$$L(\partial) = -\frac{1}{M} \sum_{m=1}^M \sum_{n=1}^N \varphi(Y^m = n) \log(Y^m = n | y^m; \partial). \quad (3)$$

Here, M represents the number of pixels, N represents the number of classes, y represents the input pixel, and $\varphi(\cdot)$ is an indicator function. If $m = n$, the loss equals 1; otherwise, it equals 0.

3.3. Feature Fusion Module. The feature fusion module aims to synthesize forgery object and pixel features, as shown in Figure 2. Following [15], we utilize a decoder to reconstruct the fusion features and divide the manipulation area to replace the fully connected layer. We utilize multichannel filters to generate heatmaps for manipulating images for the convolutional operation. Each decoder upsamples the feature maps discovered in the previous layer and performs convolution and batch normalization operations. We employ a 3×3 size kernel [5] for the decoder and obtain 64 and 16 feature maps in the first and second layers, respectively. The output of the feature fusion module is a heatmap $Z_{Fusion1}$ containing manipulation areas, as shown in Figure 3.

3.4. Integrated Fusion Module. The heatmaps generated by the pixel-level detection branch may contain many noisy areas, such as the blue circles in Figure 3. Generally, there are

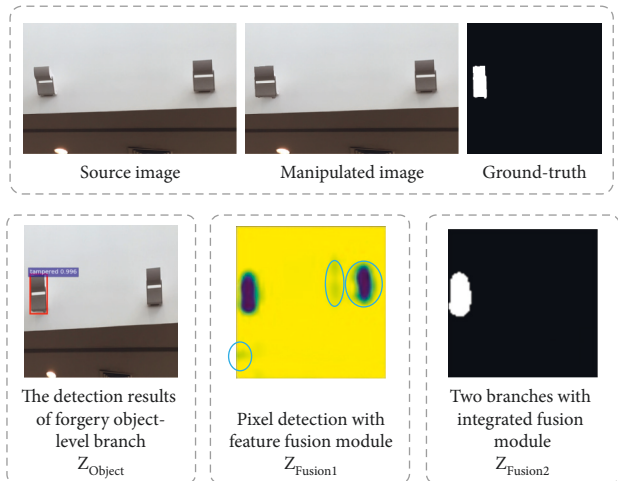


FIGURE 3: The outputs of our approach. The first row indicates the source image, the manipulated images, and the ground truth. The second row demonstrates the detection result of forgery object-level branch (Z_{Object}), the pixel-level branch with the feature fusion module ($Z_{Fusion1}$), and the two branches with the integrated fusion module ($Z_{Fusion2}$).

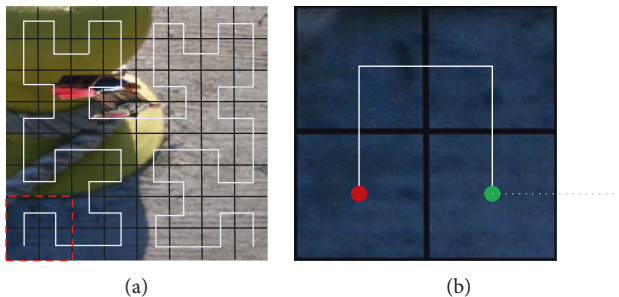


FIGURE 4: How the Hilbert curve works in an image.

two typical fusion methods when fusing Z_{Object} and $Z_{Fusion1}$ to $Z_{Fusion2}$, i.e., the AND operation and OR operation.

$$Z_{Fusion2} = Z_{Fusion1} \circledast Z_{Object}. \quad (4)$$

For the AND operation, we keep the heatmap in the bounding box from $Z_{Fusion1}$. Meanwhile, the heatmap areas that are out of the bounding box are ignored, as shown in Figure 5(a). In the OR operation, we preserve all the areas in the bounding box and the heatmaps, as shown in Figure 5(b).

We devise a multiresolution fusion (MRF) strategy based on the bounding box from both the pixel and object levels, as shown in Figure 5(c). We follow the AND operation if the bounding box in the forgery object detection branch encompasses the pixel-level detection results. Meanwhile, we take all the areas in the bounding box if there are no pixel-level detection results, as described in Algorithm 1.

4. Experiments

4.1. Experimental Datasets and Evaluation Metrics. Implementation Details. We implement our proposed approach in TensorFlow. We utilize two NVIDIA Quadro RTX

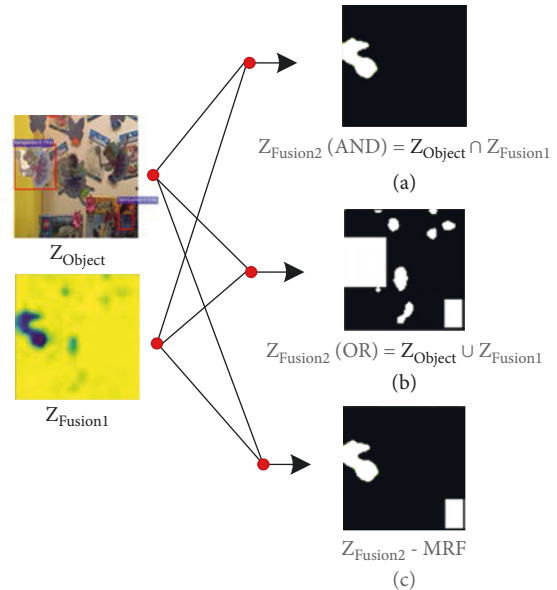


FIGURE 5: Using integrated fusion method to choose the manipulation area.

5000 GPUs to expedite our computational load. We set the batch size to 16. The learning rate is set to 0.00003 in pixel level. We set 0.001 as the initial value in object level and then reduce it to 0.0001 after 40k steps.

Datasets: we compared our method with current state-of-the-art methods on NIST Nimble 2016 (NIST'16) [16], CASIA [17, 18], Coverage [19], and Columbia [20].

- (1) NIST'16 dataset was released in 2016, and it is a standard dataset for image manipulation detection. It covers the three types of manipulations: splicing, copy-move, and removal.
- (2) CASIA dataset was released in 2013, and it covers two types of manipulations: splicing and copy-move. Some postprocessing is used in manipulation regions, like filtering and blurring, to improve the difficulty of detection.
- (3) Coverage dataset was released in 2016, and it covers six types of copy-move manipulation, such as copy only and shape change. The dataset includes the source and mask images and has a similarity measure for manipulating images.
- (4) Columbia dataset focuses on splicing based on uncompressed images.

Evaluation metric: we utilize the F1 score in pixel- and object-level under the area under curve (AUC) as our evaluation metrics for performance comparison.

- (1) F1 score is a pixel-level evaluation index for image manipulation detection, and it is used to estimate the similarity between predicted results and actual value.
- (2) AUC is the area under the ROC curve, an essential indicator for measuring detection accuracy. Based on the intersections between ROC curves, we can evaluate the consequence of the model.

Input: Bounding boxes in the forgery object detection branch: $B_i, i = 1, 2, \dots, N$. N : The number of bounding boxes. Pixel-level detection results in pixel-level detection branch: $P_j, j = 1, 2, \dots, M$. M : The number of results.

Output: forgery areas: $F_p, p = 1, 2, \dots, N$

assign Intersection $_{ij}$ to \bar{B}_i AND P_j

for each B_i do

 if Intersection $_{ij}$ is not null:

 assign Intersection $_{ij}$ to F_p

 else

 assign B_i to F_p

return F_p

ALGORITHM 1: The preprocessing of the multi-resolution fusion (MRF) strategy.

Baseline models: we compare our approach with different baseline models.

- (1) ELA [21]: this approach detects quality loss caused by JPEG compression by calculating the error between the actual and manipulation areas.
- (2) NOI1 [22]: this approach simulates local noise through wavelet coefficients and utilizes the inconsistency of the noise to detect the manipulation area.
- (3) CFA1 [23]: this approach is based on the CFA estimation method. It employs adjacent pixels to simulate the filter array image from the camera and calculate the manipulation probability for each pixel.
- (4) MFCN [2]: this approach is based on an edge-enhanced multitask complete convolutional network. Moreover, it is used to detect manipulation by predicting the area edge.
- (5) J-LSTM [24]: this approach is based on the LSTM network, which judges the pixel-level manipulations by separating the image into blocks.
- (6) RGB-N [7]: this approach is based on Faster R-CNN to establish a model for RGB and noise streams.
- (7) LSTM-Encoder [5]: this approach employs a hybrid CNN-LSTM model to detect manipulation regions.
- (8) C2RNet [8]: this approach includes C-CNN and R-CNN to distinguish the genuine and manipulation images.

Pretrained model: we train the forgery object-level and pixel-level branch separately. The forgery object-level branch, which needs to set the bounding box as the object label by the frame around the pixel-level ground truth, is trained first. Then, we train the pixel branch and employ the features extracted by the forgery object encoder to train the feature fusion module. Finally, we combine the results in an integrated fusion module using logical operations in Section 3.4.

We utilize the synthetic dataset created by Bappy et al. [5] in the pixel-level branch using the DRESDEN, COCO, and NIST'16 datasets. We follow [7] to set up the forgery object-level branch. Moreover, we utilize ResNet-101 in

Faster R-CNN, which is pretrained on ImageNet, to extract the features. We train the pixel branch by using 90% of the images for training and 10% for validation.

4.2. Experimental Analysis. Experiment preparation: we test our proposed method on four datasets, NIST'16 [16], CASIA [17, 18], Coverage [19], and Columbia. Table 1 shows the comparison results of the pixel-level F1 score and AUC. We compare four sets of experiments. The first column and the second column are the results of the single pixel-level branch and single forgery object-level branch. The third column is the result of the double branch, which includes the AND operation, OR operation, and the MRF proposed in Section 3.4.

The MRF has a better performance, as shown in Table 1. Therefore, we choose the MRF as the follow-up experimental approach and set 0.2 as the manipulated threshold in the heatmap.

Result analysis: Table 2 lists the F1 score comparison between our method and the baselines. Table 3 provides the AUC comparison. We utilize the experimental results from [7, 13] and [5].

As shown in Table 2, the F1 score is a classification accuracy metric that combines precision and recall, which means the larger the F1 score is, the more robust the model is. The CASIA dataset has postprocessing methods, which affect our forgery object-level branch in detected forged objects. In comparison to our method, the LSTM-encoder [5] approach utilizes pixel-level detection and focuses on spatial cues. As a result, the LSTM-encoder [5] approach's F1 score is 0.3% higher than ours on CASIA. However, our approach combines both object- and pixel-level branches. Using our model, the F1 score on the other datasets is improved and the detection of postprocessing methods is considered. As a result, our approach performs better than other methods on the NIST'16 [16], Coverage [19], and Columbia [20] datasets.

Our approach outperforms the baselines on the CASIA, Coverage, and Columbia datasets for the pixel-level AUC comparison. Especially on the Coverage dataset, our approach has 1.1% improvement compared to the second-best

TABLE 1: The pixel-level F1 score/AUC comparison on four standard datasets.

	Pixel-level		Forgery object-level		“AND”		Dual branch “OR”		“MRF”	
	F1	AUC	F1	AUC	F1	AUC	F1	AUC	F1	AUC
NIST’16 [16]	0.780	0.796	0.717	0.912	0.739	0.854	0.828	0.892	0.807	0.929
CASIA [17, 18]	0.751	0.713	0.688	0.844	0.744	0.743	0.742	0.851	0.789	0.862
Coverage [19]	0.551	0.723	0.433	0.802	0.531	0.808	0.561	0.828	0.563	0.828
Columbia [20]	0.831	0.641	0.398	0.756	0.529	0.809	0.776	0.803	0.833	0.801

TABLE 2: The pixel-level F1 score comparison on the standard datasets.

	NIST’16 [16]	CASIA [17, 18]	Coverage [19]	Columbia [20]
ELA [21]	0.236	0.470	0.222	0.214
NOI1 [22]	0.285	0.574	0.269	0.263
CFA1 [23]	0.174	0.467	0.190	0.207
MFCN [2]	0.571	0.612	-	0.541
RGB-N [7]	0.722	0.697	0.437	0.408
LSTM-encoder [5]	0.789	0.792	—	0.823
C2RNet [8]	0.55	0.676	—	0.695
Ours	0.807	0.789	0.563	0.833

TABLE 3: The pixel-level AUC comparison on the standard datasets.

	NIST’16 [16]	CASIA [17, 18]	Coverage [19]	Columbia [20]
ELA [21]	0.429	0.581	0.583	0.613
NOI1 [22]	0.487	0.546	0.578	0.612
CFA1 [23]	0.501	0.720	0.485	0.522
J-LSTM [24]	0.764	—	0.614	—
RGB-N [7]	0.937	0.858	0.817	0.795
LSTM-encoder [5]	0.794	—	0.712	—
Ours	0.929	0.862	0.828	0.801

result. Moreover, RGB-N [7] outperforms our approach on the NIST’16 dataset. This is mainly because the forged image quality is low in NIST’16. The pixel-level branch has inaccurate pixel classification, leading to poor boundary box regression. However, RGN-N only uses forgery object-level information and performs better on NIST’16. Therefore, the manipulation of object detection at different scales is the next point of study.

Visualization results: we illustrate some visualization results in Figures 6 and 7 in comparison with the pixel-level, forgery object-level, and the dual branch (MRF). Images are selected from Coverage and NIST’16. We illustrate better results in Figure 6. As we can see, our approach can detect image manipulation accurately. The two branches can correct for each other. The pixel-level branch can segment the forged objects from the bounding box detected by the forgery object-level branch, which makes the results finer-

grained, such as Line 1 in Figure 6. Meanwhile, the forgery object-level branch shields the noise points of the pixel-level branch and obtains better performance, such as Line 4 in Figure 6.

Meanwhile, in Figure 7, we select a few poor cases, in which the pixel branch causes the result in the first row, and the forgery object-level branch causes the result in the second row. Similar to Line 1 in Figure 7, the forgery object-level branch detects the manipulated region precisely, but the pixel-level branch does not detect the whole manipulated region, which causes the poor result. Similarly, in Line 2, the pixel-level branch successfully detects part of the image manipulation area, but the forgery object-level branch does not detect the bounding box. Both situations lead to detection failures. We will balance the detection results of the two branches as much as possible in future work to obtain better experimental results.

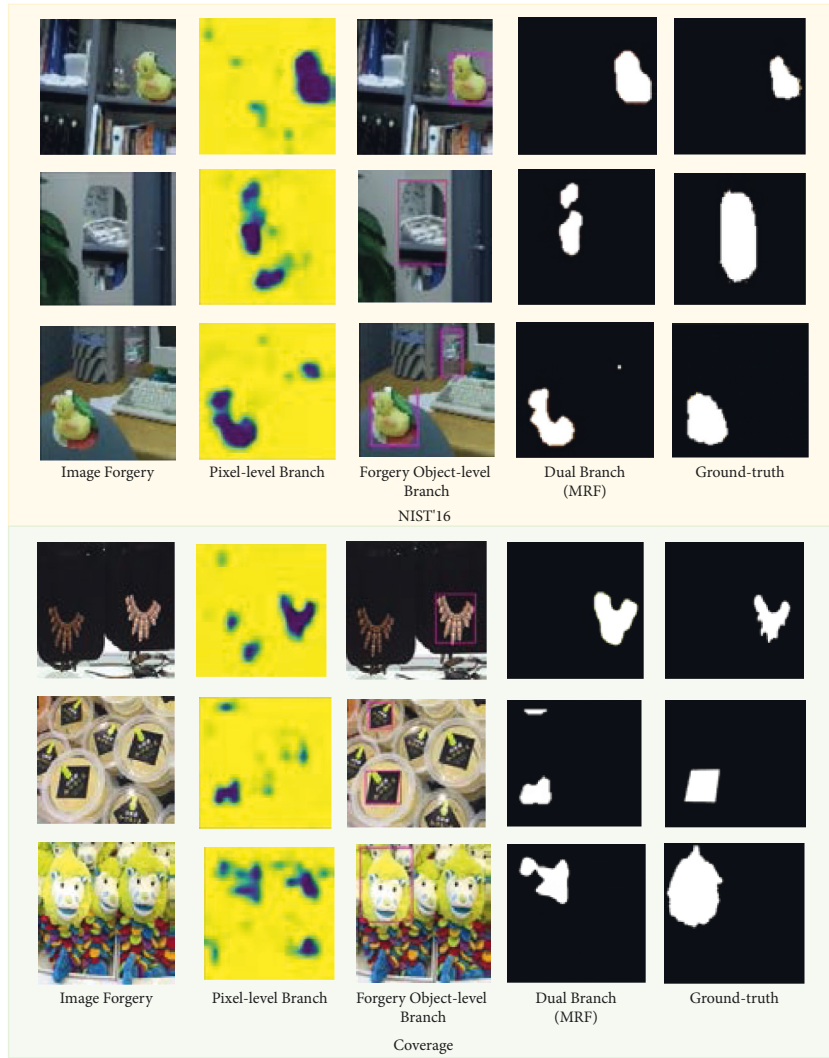


FIGURE 6: Visualization results from our approach.

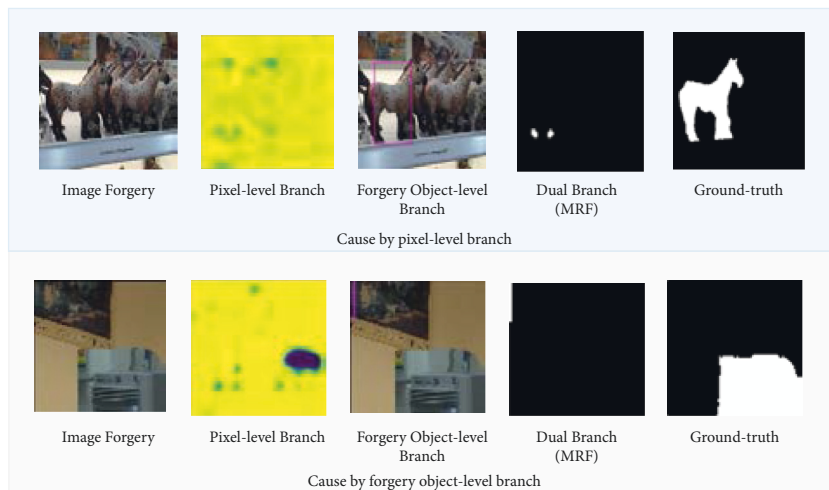


FIGURE 7: Poor visualization results from our approach.

5. Conclusion

We present a framework for image manipulation detection that combines a forgery object-level branch, a pixel-level branch, and two fusion modules. We utilize Faster R-CNN to detect manipulation areas on the forgery object-level branch. Meanwhile, we extract the resampling feature for each patch and utilize the Hilbert curve and LSTM network to detect the manipulated regions in the pixel-level branch. We fuse the two branches with the fusion modules and obtain a binary map of the manipulation regions. Experimental results show superior performance compared with the state-of-the-art methods. However, we also find that the two branches can affect each other when the manipulated object is of low quality. We will balance the two branches as much as possible in future work to obtain better experimental results.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the Basic Scientific Research Operating Expenses of the Academy of Broadcasting Science, NRTA (No. JBKY2020006).

References

- [1] Y. Zhang, J. Goh, L. L. Win, and V. Thing, "Image Region Forgery Detection: A Deep Learning Approach," *Proceedings of the Singapore Cyber-Security Conference (SG-CRC)*, vol. 2016, Singapore, 2016.
- [2] R. Salloum, Y. Ren, and C.-C. Jay Kuo, "Image splicing localization using a multi-task fully convolutional network (MFCN)," *Journal of Visual Communication and Image Representation*, vol. 51, pp. 201–209, 2018.
- [3] B. Chen, W. Tan, G. Coatrieux, Y. Zheng, and Y. -Q. Shi, "A serial image copy-move forgery localization scheme with source/target distinguishment," *IEEE Transactions on Multimedia*, vol. 23, pp. 3506–3517, 2020.
- [4] J. Bunk, J. H. Bappy, T. M. Mohammed, A. Flenner, and B. S. Manjunath, "Detection and Localization of Image Forgeries Using Resampling Features and Deep learning," in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 1881–1889, IEEE, Honolulu, HI, USA, July 2017.
- [5] J. H. Bappy, C. Simons, L. Nataraj, B. S. Manjunath, and A. K. Roy-Chowdhury, "Hybrid LSTM and encoder-decoder architecture for detection of image forgeries," *IEEE Transactions on Image Processing*, vol. 28, no. 7, pp. 3286–3300, 2019.
- [6] J. Park, D. Cho, W. Ahn, and H. K. Lee, "Double JPEG detection in mixed JPEG quality factors using deep convolutional neural network Computer Vision (ECCV)," 2018.
- [7] P. Zhou, X. Han, V. I. Morariu, and L. S. Davis, "Learning rich features for image manipulation detection Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition," pp. 1053–1061, Salt Lake City, UT, USA, June 2018.
- [8] B. Xiao, Y. Wei, X. Bi, W. Li, and J. Ma, "Image splicing forgery detection combining coarse to refined convolutional neural network and adaptive clustering," *Information Sciences*, vol. 511, pp. 172–191, 2020.
- [9] B. Chen, X. Liu, Y. Zheng, G. Zhao, and Y. -Q. Shi, "A robust GAN-generated face detection method based on dual-color spaces and an improved Xception," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, pp. 3506–3517, 2021.
- [10] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," *Advances in Neural Information Processing Systems*, vol. 28, 2015.
- [11] Y. Wu, W. Abd-Almageed, and P. Natarajan, "Image Copy-Move Forgery Detection via an End-To-End Deep Neural network," in *Proceedings of the 2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1907–1915, IEEE, Lake Tahoe, NV, USA, March 2018.
- [12] T. M. Mohammed, J. Bunk, L. Nataraj et al., "Boosting image forgery detection using resampling features and copy-move analysis," *Electronic Imaging*, vol. 30, no. 7, pp. 118-1–118-7, 2018.
- [13] A. Mazumdar and P. K. Bora, "Two-stream Encoder-Decoder Network for Localizing Image Forgeries," 2020, <https://arxiv.org/pdf/2009.12881.pdf>.
- [14] B. Mahdian and S. Saic, "Blind authentication using periodic properties of interpolation," *IEEE Transactions on Information Forensics and Security*, vol. 3, no. 3, pp. 529–538, 2008.
- [15] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.
- [16] NIST, "Nist nimble 2016 datasets," 2016, <https://www.nist.gov/itl/iad/mig/%20nimble-challenge-2017-evaluation/>.
- [17] J. Dong, W. Wang, and T. Tan, "Casia image tampering detection evaluation database," 2010, <http://forensics.idealtest.org>.
- [18] J. Dong, W. Wang, and T. Tan, "Casia Image Tampering Detection Evaluation database," in *Proceedings of the 2013 IEEE China Summit and International Conference on Signal and Information Processing*, pp. 422–426, IEEE, Beijing, China, July 2013.
- [19] B. Wen, Y. Zhu, R. Subramanian, T. Ng, X. Shen, and S. Winkler, "COVERAGE-A Novel Database for Copy-Move Forgery detection," in *Proceedings of the 2016 IEEE International Conference on Image Processing (ICIP)*, pp. 161–165, IEEE, Phoenix, AZ, USA, September 2016.
- [20] T. T. Ng, S. F. Chang, and Q. Sun, "A Data Set of Authentic and Spliced Image blocks," ADVENT Technical Report, pp. 203–2004, Columbia University, Columbia, USA, 2004.

- [21] N. Krawetz and H. F. Solutions, "A picture's worth," *Hacker Factor Solutions*, vol. 6, no. 2, p. 2, 2007.
- [22] B. Mahdian and S. Saic, "Using noise inconsistencies for blind image forensics," *Image and Vision Computing*, vol. 27, no. 10, pp. 1497–1503, 2009.
- [23] P. Ferrara, T. Bianchi, A. De Rosa, and A. Piva, "Image forgery localization via fine-grained analysis of CFA artifacts," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 5, pp. 1566–1577, 2012.
- [24] J. H. Bappy, A. K. Roy-Chowdhury, J. Bunk, L. Nataraj, and B. S. Manjunath, "Exploiting spatial structure for localizing manipulated image regions," in *Proceedings of the IEEE international conference on computer vision*, pp. 4970–4979, Venice, Italy, October 2017.

Research Article

Domain Transferred Image Recognition via Generative Adversarial Network

Haoqi Hu , Sheng Li , Zhenxing Qian , and Xinpeng Zhang 

Fudan University, Shanghai, China

Correspondence should be addressed to Zhenxing Qian; zxqian@fudan.edu.cn

Received 23 February 2022; Accepted 5 April 2022; Published 26 April 2022

Academic Editor: Beijing Chen

Copyright © 2022 Haoqi Hu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Recent studies have demonstrated that neural networks exhibit excellent performance in information hiding and image domain transfer. Considering the tremendous progress that deep learning has made in image recognition, we explore whether neural networks can recognize the imperceptible image in the transferred domain. Our target is to transfer natural images into images that belong to a different domain, while at the same time, the attribute of natural images can be recognized on domain transferred images directly. To address this issue, we proposed domain transferred image recognition to achieve image recognition directly on the transferred images without the original images. In our proposed system, a generator is designed for the domain transfer and a recognizer is responsible for image recognition. To be flexible for the natural image restoration in some cases, we also incorporate an additional generator in our method. In addition, a discriminator will play an indispensable role in the image domain transfer. Finally, we demonstrate that our method can successfully identify the natural images on transferred images without access to original images.

1. Introduction

Recently, there have emerged numerous methods for privacy protection-awareness [1–3]. At the same time, deep learning has made great breakthroughs in speech, image, and text recognition [4–6]. However, training these networks requires a large amount of data, which makes some giant companies such as Google, Microsoft, and Amazon or personalized customization organizations which try every means to collect personal data of their users for training deep models [7]. Despite of great performance of these well-trained deep networks, they bring huge privacy risks [8, 9].

Most of small businesses or individuals use the cloud services provided by giant companies for deep learning tasks since they are limited to the local storage capacity and GPU resources. However, the data collected by these organizations can be reused repeatedly, making users difficult to delete. Besides, these sensitive data may contain unique personal identical information such as faces and voices, which inevitably bring risks when stolen by malicious attackers and used for illegal benefits [10, 11].

To securely perform image recognition, information hiding, which conceals important secret information in the

carrier (image, video, audio, etc.), can solve the issue of privacy leakage elegantly and flexibly [12, 13]. However, information hiding mainly focuses on protecting secret information from being leaked during transmission. As shown in Figure 1, when image recognition is required, it has to restore the secret images, which increases the risk of information leakage. Moreover, to hide abundant secret information, it is necessary to select an appropriate carrier with large redundant room, which is time-consuming and will inevitably increase the risk of information leakage once the carrier is intercepted. To this end, we propose a method that can achieve image recognition in the transferred domain directly, which can recognize the attributes of secret images on transferred images.

Specially, our detailed application scenario for real world is shown in Figure 2; a giant company deploys an image recognition service in the cloud for profits. To avoid data leakage of users, we train a series of models ahead to reach this goal. In the real service deployment, the service provider owns/deploys only the well-trained recognizer in its cloud, and the domain transfer generator is deployed in a trustworthy party.

Therefore, a user uploads a secret image to the trustworthy party to transfer his/her image into another image

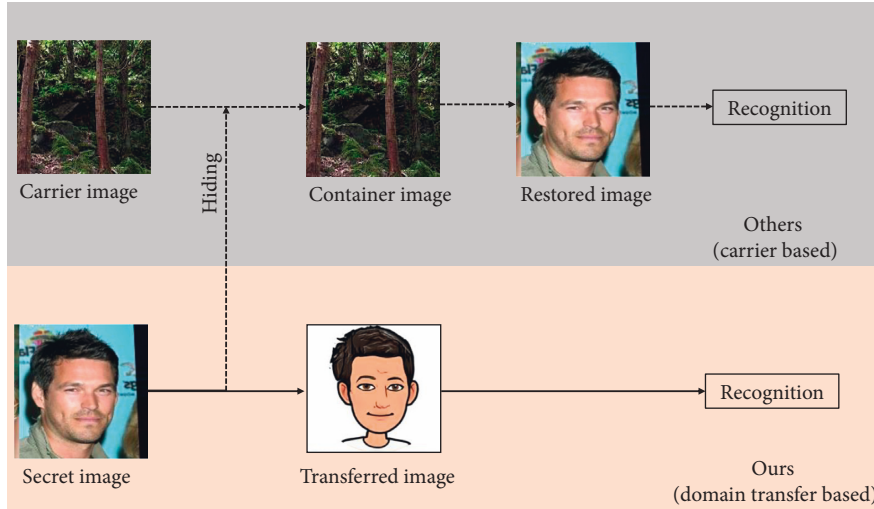


FIGURE 1: The comparison of application processes. For a system that requires image recognition in the cloud server, to protect the image content privacy, other methods based on information hiding will first select a textured carrier/cover image and then apply a special embedding algorithm to hide the secret image in a carrier image and send it to the cloud end. The receiver will apply the corresponding extracting method to obtain the secret image for recognition. Differently, our method is free from carrier image by transferring the secret image to another domain and the recognition process can be performed on the transferred domain directly.

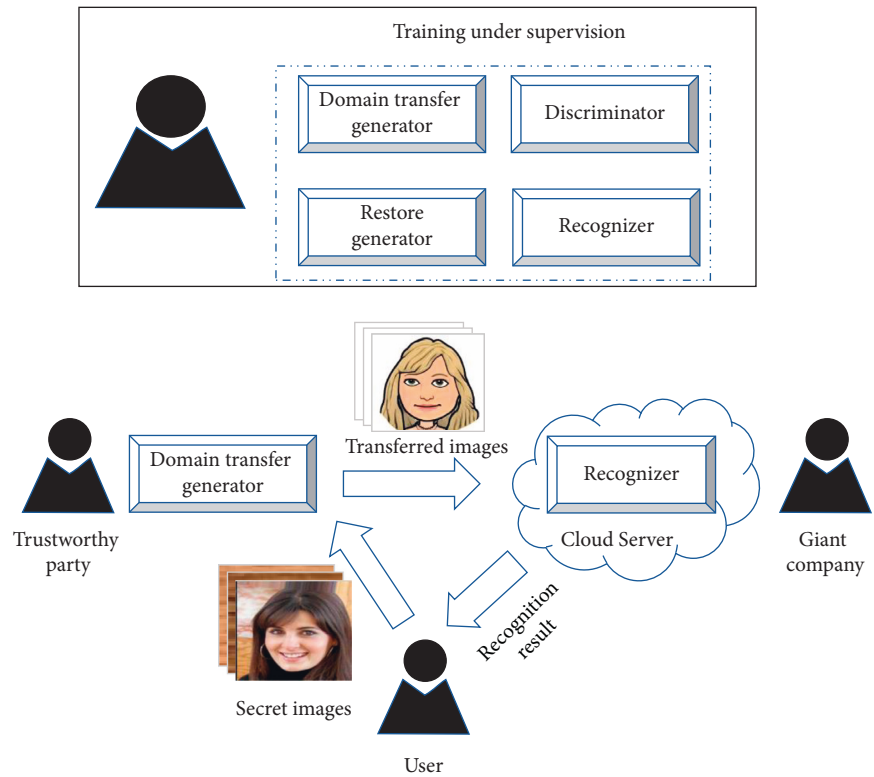


FIGURE 2: The conception of our system. The models are trained under supervision, where the domain transfer generator is used to protect the content privacy of the user’s secret images and a recognizer network which is used to classify secret images in the transferred domain. In the cloud recognition application scenario, the domain transfer generator is handed over to a trusted party, and the recognizer is controlled by an organization which is deployed on a cloud service.

which has different content/style and then delivers the transferred image to the cloud for image recognition.

To perform image recognition in the transferred domain, we designed a transfer generator to accomplish the process of domain transfer. Same as the classic generative adversarial

networks, an indispensable discriminator aids the generator to generate a high-quality image. Besides, a classifier is responsible for the image recognition task. Although our system is able to recognize the attributes of the original images through the transferred images directly, to facilitate

the original image restoration in some cases, we also incorporate an additional restore generator to recover the secret image in our method. In summary, our contributions are three-fold:

- (i) To achieve image recognition as well as avoid private information leakage issue, we proposed a framework to perform image recognition directly on domain transferred images
- (ii) Our image recognition method is performed on the transferred domain, which decreases the exposure risk of the source image and omits the process of carrier selection
- (iii) Experiments demonstrate the availability of our method in terms of classification accuracy and visual effect of the transferred images

2. Related Work

In this section, we will first review some secure inference and visual information protection strategies.

2.1. Encryption. Encryption technique is used for data privacy protection. Its basic target is to hide the content of the data, which encrypts the raw data into the ciphertext data and decrypts it back to the original version with the corresponding decryption algorithm.

Homomorphic encryption proposed by Craig [14] can support arbitrary computations on the encrypted data and the final calculation result can be obtained after decryption. However, this technique, which is calculated in the encrypted domain, is computationally expensive compared to plaintext calculation. For example, recently, Sanyal [15] proposed a homomorphic encryption-based image classification algorithm to protect image privacy. However, it takes nearly 2 hours to classify encrypted MNIST images on a 16-core workstation, which is impractical in reality. Moreover, the encrypted images can raise the awareness of attackers due to its unreasonable ciphertext.

Secure multiparty computation is an important branch of cryptography, which aims to solve the problem of collaborative computing that protects privacy among a group of untrusted parties. Implementations of predicting encrypted data based on secure multiparty computing have flourished [16, 17]. However, these methods require the data owner to encrypt the inputs and constantly interact and communicate with each other. Besides, as mentioned in [18], most of the existing multiparty computation-based secure inferences rely on customized protocols that are highly optimized for particular activation functions. For example, XONN [19] is currently the most efficient solution for 2-party protocol, but XONN only works with Sign as the activation function. Implementing exponential function (Sigmoid) or max function (ReLU) requires heavy computations and communications in SMPC-based solutions.

2.2. Information Hiding. Information hiding is one of the most important ways to protect secret data, which has been

well researched in the past decades [20–22]. This technique can be roughly classified into two categories: digital watermarking and steganography. Recently, many deep learning algorithms related to information hiding have been developed. Usually, digital watermarking technology hides a particular bit string in inconspicuous places to protect the copyright of images, models, etc. Uchida et al. [23] embed a watermark in model parameters using a regularizer to protect the intellectual property of trained models, with the performance of trained model hardly affected. Rouhani et al. [24] embed watermarking in the weight distribution of convolutional layers in trained models to protect deep learning models. Baluja [25] successfully hides a full-size color image into another image of the same size based on a deep image encoder and decoder network to realize image steganography. However, these methods focus on protecting the security of hidden information during transmission. These methods have to go through the process of extracting hidden secret information when using these information, which may lead to hidden secret information leakage after restoration.

2.3. Image Domain Transfer. Image domain transfer refers to the process of generating another image according to one image, that is, transfer one image from one domain to another. Pix2pix proposed by Isola et al. [26] provides a concise and elegant general framework for solving a series of image domain transfer tasks, and the author proves that the method has good performance in tasks such as segmentation map to street view map, grayscale map to color map, and clothing outline sketch to color map. In addition to supervised image domain tasks (with paired training samples), for unsupervised image domain transfer tasks without paired images as training samples, Zhu et al. [27] designed CycleGAN to complete the image domain transfer process from one dataset to another dataset. Moreover, StarGAN [28] proposed by Choi et al. can complete various attribute conversions of face images, such as gender, age, skin color, and emotions, and Tang et al. [29] proposed a method that can transfer an image from a source to a target domain guided by controllable structures. However, these GAN-based image domain transfer methods simply focus on approximating the distribution of generated images with the distribution of the target domain. Inspired by the success of deep learning in multitasking [30, 31], we turn our attention to the image recognition task in the transferred domain that is free from image content exposure.

3. Domain Transferred Image Recognition

In order to transfer the secret image from the source domain to a target domain, an image generator is indispensable. Besides, a necessary classifier/recognizer will be responsible for image recognition in the transferred domain. As mentioned before, the domain transfer will alter the distribution of source images, in which way the privacy of images can be protected. However, changes in the distribution of data will hamper the image recognition task. Therefore, the generator

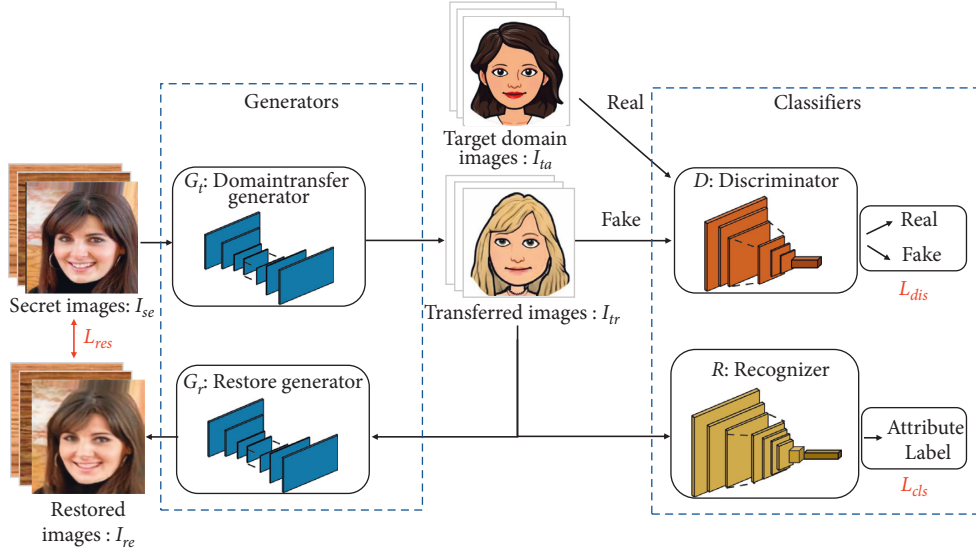


FIGURE 3: The schematic structure of our proposed system. For a secret image, it is firstly transferred to another image with a different style using the domain transfer generator. The transferred image serves as the input of the discriminator, recognizer, and restore generator. The recognizer can identify the attributes of the secret images on the transferred images. The restore generator is responsible for reconstructing the secret images based on transferred images. The discriminator aims to distinguish images in the target domain from images generated by the transfer generator.

and classifier modules will be trained in turn, just as the way of training a generative adversarial network. Besides, an alternative restore generator can also be incorporated to restore the secret image in some cases. Therefore, as shown in Figure 3, our whole framework mainly contains four modules: two generators, one discriminator, and one recognizer.

In order to describe the proposed method more clearly, we define the relevant concepts and variables as follows. The secret image dataset/domain is denoted as $\mathcal{X}_{\text{natural}}$, where the image in it is represented by I_{se} . Similarly, the target dataset/domain is defined as $\mathcal{X}_{\text{target}}$, where the image in it is denoted as I_{ta} . I_{tr} and I_{re} respectively indicate the transferred images output by the domain transfer generator and the restored images output by the restore generator. G_t and G_r represent the generator for the domain transfer and the generator for original secret image restoration, respectively. The discriminator is denoted by \mathcal{D} , and the recognizer \mathcal{R} can identify secret image attributes on the transferred domain images I_{tr} .

3.1. Domain Transfer Generator. Given a secret image I_{se} from the dataset $\mathcal{X}_{\text{natural}}$ to be identified, the domain transfer generator (corresponding to G_t in Figure 3) aims to transfer the secret image into an image in the target domain with a different style such as a cartoon face style and an animal face style. In other words, the transfer generator G_t should extract the feature of the secret image as much as possible and transform this feature into an image that conforms to the distribution of the target domain. Besides, the transferred image must fool the discriminator to make it fail to tell whether this image is generated by G_t or comes from $\mathcal{X}_{\text{target}}$. When the whole system is fully trained, the generator G_t can

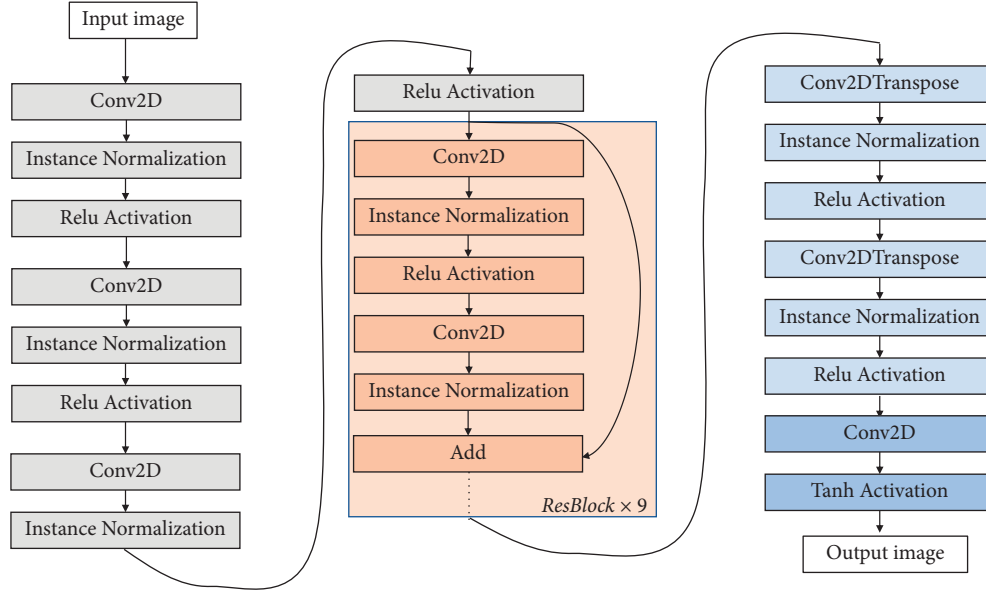
be used as an independent module to perform image transfer from $\mathcal{X}_{\text{natural}}$ domain to $\mathcal{X}_{\text{target}}$ domain.

The domain transfer process can be written as

$$I_{tr} = G_t(I_{se}). \quad (1)$$

As for the design of our domain transfer generator, given that the convolutional neural network is experimentally proven to have excellent feature learning and extraction abilities for images and that a large amount of autoencoders have shown extraordinary performance in image generation, we adopt the combination of convolutional neural network and autoencoder and design the generator G_t with a ResNet [32] structure. Specifically, as shown in Figure 4, we first stack 3 sets of convolutional layers, instance normalization layers, and ReLU activation, followed by 9 residual blocks, with each residual block containing two convolutional layers, two instance normalization layers, and one ReLU activation. The input of each residual block and the output of the latter instance normalization will be added as the output of each residual block. Finally, 2 sets of deconvolution layers, ReLU activation layers, and instance normalization layers are followed to make the size of the final generated image consistent with that of the original image. By the way, the last activation layer will be tanh to cover the full pixel range.

3.2. Discriminator. For deep models based on generative adversarial networks, the discriminator \mathcal{D} is indispensable in ensuring the quality of the generated images. The discriminator is expected to learn features that can distinguish the image I_{ta} in the target domain from the generated image I_{tr} in the transferred domain. When the input image is from the target domain $\mathcal{X}_{\text{target}}$, the discriminator should identify it as a “real” image, and when the input image is generated by


 FIGURE 4: Structure of domain transfer generator G_t and restore generator G_r .

the generator G_t , the discriminator should identify it as a “fake” image, which can be mathematically expressed as

$$t = \mathcal{D}(I_{in}), \quad (2)$$

where t represents the output of discriminator \mathcal{D} , $t \in \{\text{“real”}; \text{“fake”}\}$: image from target domain, “fake”: image generated by G_t , and I_{in} represents the input image of \mathcal{D} .

As an auxiliary module of the generator G_t , for each input image, the discriminator \mathcal{D} needs to judge whether it is “true” or “false.” But, unlike generators which have to perform complex image generation tasks, the discriminator only makes a binary decision. Therefore, our framework only contains one discriminator, a very “shallow” network. The specific structure is shown in Figure 5(a). First, we stack a convolutional layer and a ReLU activation layer and then go through 3 convolutional blocks, each of which contains one convolutional layer, one instance normalization layer, and one ReLU activation layer, finally appending one convolutional layer as the end of the discriminator.

3.3. Recognizer. For the recognition network \mathcal{R} , its main task is to be able to identify the attribute information of the original secret image I_{se} from the image I_{tr} generated by the domain transfer generator G_t , that is,

$$l = \mathcal{R}(I_{tr}), \quad (3)$$

where l is the predicted result.

Through the combination of several proposed modules, for a secret image I_{se} , the generator G_t firstly transforms the secret image to the target domain $\mathcal{X}_{\text{target}}$ and retains features that can characterize its attributes. Then, the recognizer \mathcal{R} is able to extract this feature from the generated image and maps it to the attribute label representing the secret image. This process can be expressed as

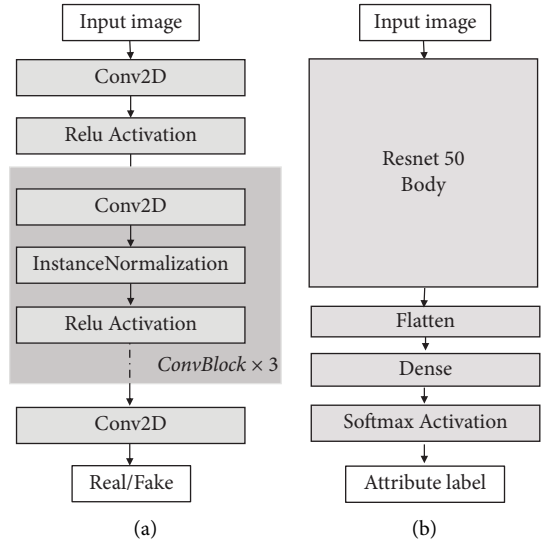


FIGURE 5: Structures of (a) discriminator and (b) recognizer.

$$\mathcal{R}: \{G_t: I_{se} \longrightarrow \mathcal{X}_{\text{target}}\} \longrightarrow l. \quad (4)$$

There are currently some popular network structures that perform very well in image recognition tasks, such as VGG [33] and ResNet [32]. Therefore, without loss of generality, we adopt ResNet structure as backbone for the recognizer in our framework and design different model heads according to the specific recognition task. Specifically, we take ResNet50 [32] as the backbone of our recognizer and replace the fully connected layer at the end of the model. The final number of output logits of the recognizer equals to the number of categories. The specific structure of the network is shown in Figure 5(b). After the input image passes through the backbone of ResNet50, the obtained features are flattened, passed through a fully connected layer, and activated by the

“softmax” function. Finally, the predicted label corresponds to the category label with the highest confidence.

3.4. Restore Generator. The recognizer designed in this paper can already recognize images, but in some scenarios that require more flexible authority certification or image processing operation, it may be necessary to recover the original secret image. Therefore, in addition to the recognizer for image recognition/authority certification, we also provide an additional restore generator G_t for image restoration. The restore generator receives the image generated by the domain transfer generator G_t as input and outputs an image as identical to the original secret image as possible. Since the task of the restore generator G_r is also to generate images, in order to reduce the complexity of the whole system, we set the structure of the restore generator G_r the same as the domain transfer generator G_t , which is shown in Figure 4.

3.5. Objective Loss Function. For a given secret image $I_{se} \in \mathcal{X}_{\text{natural}}$ and its label and an image in the target domain $I_{st} \in \mathcal{X}_{\text{target}}$, the designed network will be trained using the following adversarial loss.

For the discriminator \mathcal{D} , its input is the image I_{tr} generated by the generator or the image I_{ta} in the target domain $\mathcal{X}_{\text{target}}$, and its task is to be able to distinguish between these two kinds of images, which is a binary classification problem. In our framework, the images generated by the domain transfer generator are regarded as negative samples, and the images from the target domain are regarded as positive samples. Therefore, the discriminator will generate the following loss:

$$\mathcal{L}_{dis} = \mathbb{E}_{I_{st}} [\log \mathcal{D}(I_{st})] + \mathbb{E}_{I_{se}} [\log (1 - \mathcal{D}(G(I_{se})))] . \quad (5)$$

For the recognizer \mathcal{R} , its purpose is to predict the attribute label of the secret image. Therefore, the cross-entropy loss function, most frequently used in image recognition, urges the recognizer to make correct prediction to the target label:

$$\mathcal{L}_{cls} = - \sum_{i=1}^K y_i \log(\mathcal{R}(I_{tr})), \quad (6)$$

where K is the number of image categories, y is the image label, and $\mathcal{R}(I_{tr})$ is the predicted probability of the recognition model on the domain transferred image I_{tr} .

At the same time, the proposed framework also provides the function of restoring the original secret image. That is to say, the output of the restore generator G_r should be as same as possible to the secret image. To this end, the following loss function controls the similarity between I_{se} and I_{re} :

$$\mathcal{L}_{res} = \|I_{se} - I_{re}\|_2. \quad (7)$$

Finally, we will obtain a total loss as follows:

$$\mathcal{L}(G_r, G_t, G, \mathcal{R}) = \mathcal{L}_{dis} + \mathcal{L}_{cls} + \mathcal{L}_{res}. \quad (8)$$

4. Experiments

In this section, we will first introduce our datasets and evaluation metrics and experimental details. Then, we will

demonstrate the effectiveness of our method on varied datasets.

4.1. Dataset. Since the proposed method is to transform images in one dataset/domain to another dataset/domain, for a complete domain transfer, two datasets are required, namely, the secret image dataset/domain $\mathcal{X}_{\text{natural}}$ and the target dataset/domain $\mathcal{X}_{\text{target}}$. We select face images indicating strong privacy as our source dataset/domain, and we adopt CelebA [34] and Pubfig [35] dataset in the experiment.

CelebA [34] is a large-scale face image database containing 202,599 images with 40 categories collected from the Internet by the Chinese University of Hong Kong. Each of these images has 40 binary attributes, such as gender, attractive or not, and young or not. Without loss of generality, we use the attribute “gender” as our prediction attribute, and all the images will be resized to resolution of 256×256 . The first 10K images and the subsequent 2K images are respectively used as the training dataset and the test dataset in the experiment.

Pubfig [35] is a face image dataset with 58,797 images of 200 categories collected from the Internet. Each category has an average of 300 face images of one person. However, due to the copyright and privacy issue, most of the image links provided by the paper [35] are invalid now. As an alternative, we use the version published by other user on the Kaggle platform [36] including only 11,640 images with 150 categories. Specifically, we randomly choose 80% images as training dataset and the remain 20% images as testing dataset.

For the target dataset/domain $\mathcal{X}_{\text{target}}$, we mainly use Bitmoji-style cartoon face images. The Bitmoji [37] dataset is a cartoon style face downloaded directly from the mobile app. The Bitmoji [37] dataset contains 4085 faces with the resolution of 384×384 . In the experiments, all the Bitmoji images are resized to the same size as the source domain $\mathcal{X}_{\text{natural}}$, with the resolution of 256×256 . Figure 6 illustrates some examples of the Bitmoji dataset.

4.2. Evaluation. Since our framework is to perform image recognition in the transferred domain, the accuracy of image recognition is one of our goals. At the same time, we also provide a restore generator G_r to recover the original secret image. Therefore, the PSNR and SSIM metrics, which are most commonly used in digital image processing, are used to measure the quality of the recovered image. Given a reference image \mathbf{I} and a test image \mathbf{K} , both with size $m \times n$, the PSNR between \mathbf{I} and \mathbf{K} is defined as

$$\text{PSNR} = 10 \cdot \log_{10} \left(\frac{L^2}{\text{MSE}} \right), \quad (9)$$

where L is the dynamic range of allowable image pixel intensities (usually takes 255). For our 3-channel color image, we first calculate the MSE value of each channel and then calculate the average to get the MSE in equation (9).



FIGURE 6: Examples of Bitmoji images. The Bitmoji images contain cartoon faces of different genders, different hair colors, etc., all of which are centered and surrounded by white space.

$$MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [\mathbf{I}(i, j) - \mathbf{K}(i, j)]^2. \quad (10)$$

The SSIM is given by

$$SSIM = \frac{(2\mu_{\mathbf{I}}\mu_{\mathbf{K}} + c_1)(2\sigma_{\mathbf{IK}} + c_2)}{(\mu_{\mathbf{I}}^2 + \mu_{\mathbf{K}}^2 + c_1)(\sigma_{\mathbf{I}}^2 + \sigma_{\mathbf{K}}^2 + c_1)}, \quad (11)$$

where $\mu_{\mathbf{I}}$ and $\mu_{\mathbf{K}}$ are the local means, $\sigma_{\mathbf{I}}$ and $\sigma_{\mathbf{K}}$ are the standard deviations and $\sigma_{\mathbf{IK}}$ is the cross-covariance for images \mathbf{I} and \mathbf{K} sequentially, and c_1 and c_2 are 6.50 and 58.5, respectively, by default.

4.3. Implementation Details. The implementation is based on Keras with TensorFlow as the backend. In our experiments, we use Adam [38] optimizer with a learning rate of 0.001 and linearly decay it to 0 after 50 training epochs. Our training batch size is set 4 and it takes our 4 days for training about 200 epochs on one single NVIDIA RTX 1080 Ti GPU.

4.4. Experimental Results. We first take the CelebA dataset as the domain $\mathcal{X}_{\text{natural}}$, the Bitmoji dataset as the domain $\mathcal{X}_{\text{target}}$, and the “gender” attribute in the CelebA dataset as the recognized attribute. After sufficient training, the obtained domain transferred images and the restored images are shown in Figure 7, where images in the first row belong to the CelebA dataset, images in the second row are transferred by the domain transfer generator G_t , and images in the last row are the recovered images. Visually, the transferred face image is similar to the cartoon face image in the Bitmoji dataset, which are all centered, frontal, and surrounded by white space. It is difficult to distinguish between these two kinds of images visually.

In order to verify the generalization of our method, we also use Pubfig dataset and the Bitmoji dataset as $\mathcal{X}_{\text{natural}}$ and $\mathcal{X}_{\text{target}}$ respectively for training, where the identity of Pubfig images is used as the prediction label. The results are shown in Figure 8.

4.4.1. Comparison of Other Methods. Table 1 shows the comparison of our method with other related works from various aspects. Tao et al. [39] and Baluja [25] focus on the secure secret message communication but failed in image recognition. The domain transfer methods [40–42] are free from carrier selection but none of them take secure image recognition into consideration. Our method can not only support secret image recovery and direct image recognition but also is free from carrier selection. Besides, the difference

between our secret images and transferred images is enough (middle degree) to prevent the adversary from inferring the image content.

4.4.2. Visualization of Domain Transfer. Since the proposed model needs to transfer the image from one domain to another domain for “camouflage,” the features of images obtained by domain transfer should be as close to the target domain $\mathcal{X}_{\text{target}}$ as possible. In the field of machine learning, there are some classic feature compression/dimension reduction methods, such as PCA [43], t-SNE [44], and LLE [45]. In order to explicitly portray the domain transfer process of our method, we visualize the feature distribution of the dataset $\mathcal{X}_{\text{natural}}$ before and after domain transfer and the feature distribution of target dataset $\mathcal{X}_{\text{target}}$ using t-SNE. Specifically, we first flatten all the CelebA and Bitmoji images from 256×256 to $1 \times 256 * 256 * 3$ and then directly apply the t-SNE function in the Sklearn [46] library to compress them into 1×2 and characterize these images on a 2D plane. The visualization of the dataset is shown in Figure 9. From Figure 9(a), we can see that the boundary between CelebA and Bitmoji is very obvious. But after transferring images in CelebA, the distribution of transferred CelebA and Bitmoji datasets is very similar, shown by the overlapping between the red dots and blue dots in Figure 9(b).

4.4.3. Recognition of Domain Transferred Image. In our proposed framework, after the original secret face image is transformed by the domain transfer generator G_t , the corresponding recognizer \mathcal{R} should be able to directly identify the original secret image according to the transferred image. To examine the performance of the trained recognizer \mathcal{R} , we also train a recognition network directly on the original natural face as the best recognizer for comparison. Experiments are shown in Table 2. From the table, we can see that even if the image is transferred into other domains, our image recognition accuracy hardly decreases compared to the highest untransferred image recognition accuracy. For CelebA images, the recognition accuracy drops from 92.4% \rightarrow 92.1% and Pubfig images drop from 89.7% \rightarrow 88.4%.

4.4.4. Recognition of Reconstructed Images. As mentioned before, in order to use our proposed method more flexibly in some scenarios where the original secret image needs to be recovered, we also include a restore generator G_r for recovering the original secret image, and the usage of the restored image should not be affected. In order to test the effect of domain transfer on secret images, we test the image recognition performance of restored images. Specifically, we

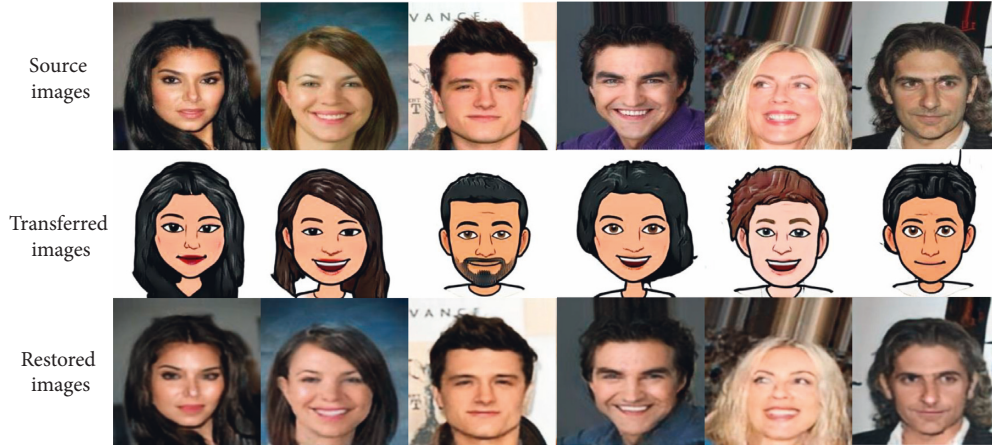


FIGURE 7: Visualization of domain transferred CelebA images and the corresponding restored images. From top to last row are original secret images I_{se} , domain transferred images I_{tr} , and recovered secret images I_{re} , respectively.

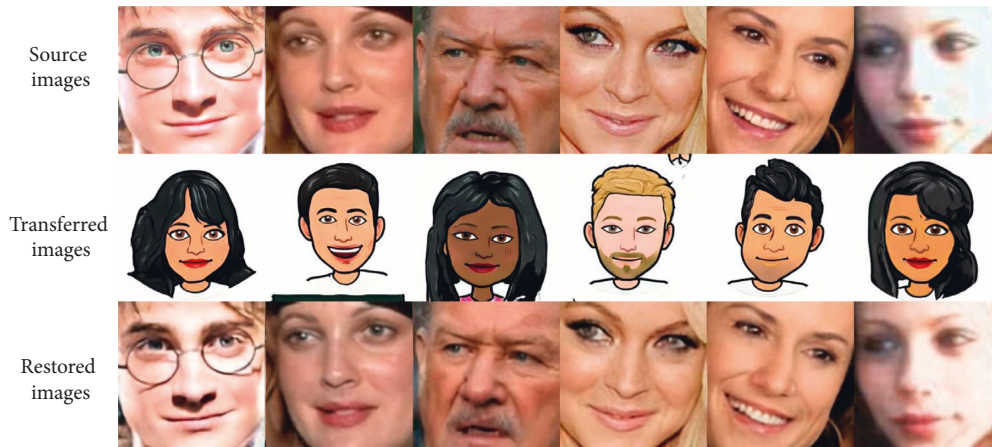


FIGURE 8: Visualization of domain transferred Pubfig images and the corresponding restored images. From top to last row are original secret images I_{se} , domain transferred images I_{tr} , and recovered secret images I_{re} , respectively.

TABLE 1: Comparison of relevant studies from different aspects. Difference means the difference between the secret image and the stego/generated/transferred image. STC means syndrome trellis coding; CNN and GAN mean convolutional neural network and generative adversarial networks, respectively.

Ref.	Carrier required	Difference	Technology adopted	Support secret images recovery	Support direct image recognition
Tao et al. [39]	Yes	Large	STC	Yes	No
Baluja [25]	Yes	Large	CNN	Yes	No
Kim et al. [40]	No	Middle	GAN	Yes	No
Chen et al. [41]	No	Middle	GAN	Yes	No
Liu et al. [42]	No	Small	GAN	No	No
Ours	No	Middle	GAN	Yes	Yes

train several recognizers on natural face images with different kinds of attributes and use these trained recognizers to test the image recognition accuracy on restored images. Without loss of generality, we select four attributes of “Male,” “Bald,” “Heavy_Makeup,” and “Attractive” on the CelebA dataset to test the recognition performance of reconstructed images. The experimental results are shown in Table 3.

As can be seen from Table 3, when the network is trained directly with the original CelebA dataset, the recognition

accuracies of “Male,” “Bald,” “Heavy_Makeup,” and “Attractive” attributes are 92.4%, 98.2%, 90.0%, and 80.2%, respectively. Even if the original image has gone through the domain transfer, the recognition accuracy of the restored image is hardly affected, with the accuracy of “Male,” “Bald,” “Heavy_Makeup,” and “Attractive” attributes being 92.2%, 98.2%, 87.7.0%, and 79.7%, respectively. Compared with the highest recognition accuracy of plaintext/baseline, the average value of the recognition accuracy has only dropped by less than one point (90.2% \rightarrow 89.5%), indicating that the

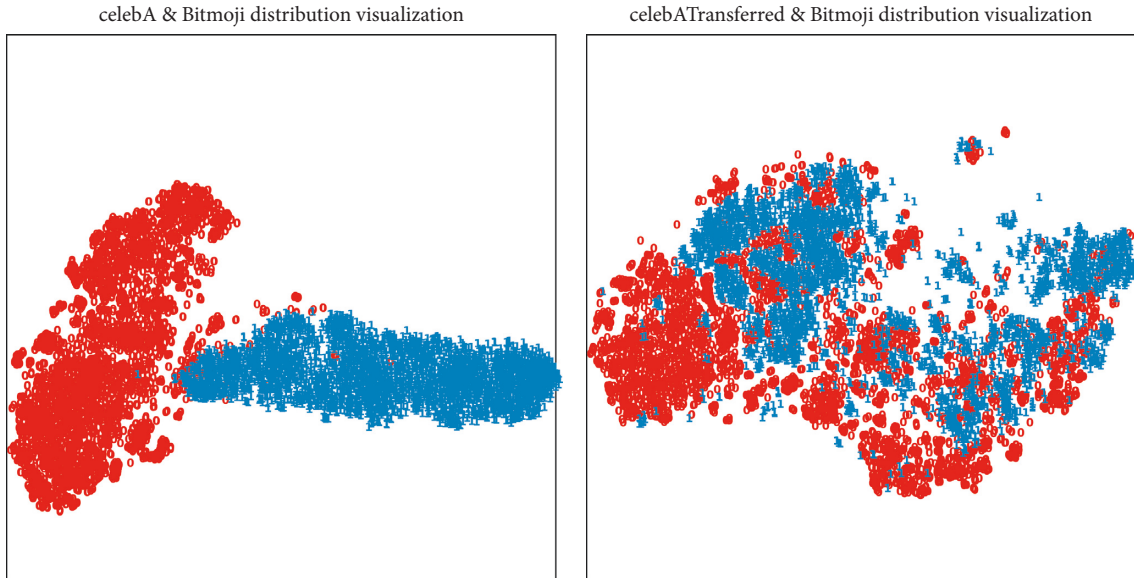


FIGURE 9: Visualization of the dataset after reducing the images to 2 dimensions using the t-SNE [44] algorithm. (a) The distribution of the CelebA dataset and Bitmoji dataset. The red dot means images in the CelebA dataset, and the blue dot means images in the Bitmoji dataset. (b) The distribution of the domain transferred CelebA dataset and Bitmoji dataset.

TABLE 2: Recognition accuracy comparison. The ACC in the “CelebA” column and “Pubfig” column means the top accuracy obtained by recognizers trained by natural, unaltered face images. The ACC in the “CelebA-T” column and “Pubfig-T” column means recognition accuracy obtained on transferred images.

Dataset	CelebA	CelebA-T	Pubfig	Pubfig-T
ACC (%)	92.4	92.1	89.7	88.4

TABLE 3: The recognition accuracy of CelebA and reconstructed CelebA images. ACC(Base) means the best accuracy obtained by the model trained on natural images. ACC(Res) means the recognition accuracy obtained on the restored images.

Attributes	Male	Bald	Heavy_Makeup	Attractive	Average
ACC(Base)	92.4	98.2	90.0	80.2	90.2
ACC(Res)	92.2	98.2	87.7	79.7	89.5

image reconstructed in our method can still be used with little performance penalty.

4.4.5. Visual Quality of Reconstructed Images. From the perspective of secret information transfer, the process of recovering the original secret image in our method can also be used for secret communication. The method of hiding images within images mentioned in [25] is an advanced method based on neural networks for large capacity information hiding, where the author pointed out that if the natural image is directly fed into the proposed neural network, the hidden image will be exposed in the residual image. To eliminate the traces of hidden image content in residual images between cover images and stego images, a simple way is to permute the pixels of hidden images before

TABLE 4: Quantitative visual quality of recovered images.

Method	Baluja [25]	Baluja (shuffled) [25]	Ours
PSNR	34.2	31.6	26.4
SSIM	0.962	0.943	0.948

they are passed to the preparation network. Following [25], we retrained our whole network to hide images without the spatial coherence of natural images. As shown in Table 4, although the average values of PSNR and SSIM of the restored images are slightly inferior to the shuffled version, the recognition of restored images will be little affected. One thing that needs to be noted is that we are only sacrificing a little bit of image quality to enable image recognition while preserving privacy. Besides, our method is free from cover image selection hence decreasing the complexity of the system during the usage/inference phase.

5. Conclusion

In this paper, we proposed a technique for image recognition while protecting the privacy of image content. First, we point out that our method is free from not only complex computation such as encryption algorithms but also carrier selection like information hiding-based method. Second, we designed and trained a combined network containing two generators, one recognizer, and one discriminator, where these two generators are responsible for domain transfer and image reconstruction, respectively, and the recognizer for image recognition on domain transferred images. Experiments are conducted on several standard datasets and the results have validated the effectiveness of our proposed method.

Data Availability

The CelebA data used to support the findings of this study are from previously reported studies and datasets, which have been cited. The Pubfig and Bitmoji data used to support the findings of this study were supplied by Kaggle under license, which can be downloaded by hyperlinks provided in references [36] and [37], respectively.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

- [1] P. Li, J. Li, Z. Huang et al., "Multi-key privacy-preserving deep learning in cloud computing," *Future Generation Computer Systems*, vol. 74, pp. 76–85, 2017.
- [2] X. Liu, L. Xie, Y. Wang et al., "Privacy and security issues in deep learning: a survey," *IEEE Access*, vol. 9, pp. 4566–4593, 2020.
- [3] Y. Qu, S. Yu, L. Gao, W. Zhou, and S. Peng, "A hybrid privacy protection scheme in cyber-physical social networks," *IEEE Transactions on Computational Social Systems*, vol. 5, no. 3, pp. 773–784, 2018.
- [4] Y. I. Daradkeh, I. Tvoroshenko, V. Gorokhovatskiy, L. A. Latiff, and N. Ahmad, "Development of effective methods for structural image recognition using the principles of data granulation and apparatus of fuzzy logic," *IEEE Access*, vol. 9, Article ID 13417, 2021.
- [5] C. Luo, L. Jin, and Z. S. Moran, "A multi-object rectified attention network for scene text recognition," *Pattern Recognition*, vol. 90, pp. 109–118, 2019.
- [6] A. B. Nassif, S. Ismail, I. Attili, M. Azzeh, and K. Shaalan, "Speech recognition using deep neural networks: a systematic review," *IEEE Access*, vol. 7, Article ID 19143, 2019.
- [7] M. Xue, C. Yuan, H. Wu, Y. Zhang, and W. Liu, "Machine learning security: threats, countermeasures, and evaluations," *IEEE Access*, vol. 8, Article ID 74720, 2020.
- [8] J. Han, W. Zang, Y. Meng, and R. Sandhu, "Quantify co-residency risks in the cloud through deep learning," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 4, pp. 1568–1579, 2020.
- [9] O.-A. Kwabena, Z. Qin, T. Zhuang, and Z. Qin, "MSCryptoNet: multi-scheme privacy-preserving deep learning in cloud computing," *IEEE Access*, vol. 7, Article ID 29344, 2019.
- [10] Z. Li, W. Xu, H. Shi, Y. Zhang, and Y. Yan, *Security and Privacy Risk Assessment of Energy Big Data in Cloud Environment*, Computational Intelligence and Neuroscience, vol. 2021, Article ID 2398460, 11 pages, 2021.
- [11] Y. Liu, Z. Ma, X. Liu, S. Ma, and K. Ren, "Privacy-preserving object detection for medical images with faster rcnn," *IEEE Transactions on Information Forensics and Security*, vol. 17, 2019.
- [12] X. Liao, Y. Yu, B. Li, Z. Li, and Q. Zheng, "A new payload partition strategy in color image steganography," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 3, pp. 685–696, 2019.
- [13] H. Zhao, Q. Dai, J. C. Ren, W. Wei, Y. Xiao, and C. Li, "Robust information hiding in low-resolution videos with quantization index modulation in dct-cs domain," *Multimedia Tools and Applications*, vol. 77, no. 14, Article ID 18827, 2018.
- [14] G. Craig, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the 40th-first annual ACM symposium on Theory of computing*, pp. 169–178, May 2009.
- [15] A. Sanyal, M. J. Kusner, A. Gascon, and V. Kanade, "Tapas: tricks to accelerate (encrypted) prediction as a service," in *Proceedings of the 35th International Conference on Machine Learning*, vol. 80, pp. 4490–4499, 2018.
- [16] B. Knott, S. Venkataraman, A. Hannun, S. Sengupta, M. Ibrahim, and L. van der Maaten, "Crypten: secure multi-party computation meets machine learning," *Advances in Neural Information Processing Systems*, vol. 34, 2021.
- [17] A.-T. Tran, T.-D. Luong, J. Karnjana, and V.-N. Huynh, "An efficient approach for privacy preserving decentralized deep learning models based on secure multi-party computation," *Neurocomputing*, vol. 422, pp. 245–262, 2021.
- [18] D. Anders, D. Escudero, and M. Keller, "Secure evaluation of quantized neural networks," *Proceedings on Privacy Enhancing Technologies*, vol. 2020, no. 4, pp. 355–375, 2020.
- [19] M. S. Riazi, M. Samragh, H. Chen, L. Kim, K. Lauter, and F. Koushanfar, "Xonn: xnor-based oblivious deep neural network inference," in *Proceedings of the 28th USENIX Conference on Security Symposium*, pp. 1501–1518, August 2019.
- [20] S. Dhawan and R. Gupta, "Analysis of various data security techniques of steganography: a survey," *Information Security Journal: A Global Perspective*, vol. 30, no. 2, pp. 63–87, 2021.
- [21] W. Shen, J. Qin, Y. Jia, H. Rong, and J. Hu, "Enabling identity-based integrity auditing and data sharing with sensitive information hiding for secure cloud storage," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 2, pp. 331–346, 2018.
- [22] Y. Yang, Z. Li, W. Xie, and Z. Zhang, "High capacity and multilevel information hiding algorithm based on pu partition modes for hevc videos," *Multimedia Tools and Applications*, vol. 78, no. 7, pp. 8423–8446, 2019.
- [23] Y. Uchida, Y. Nagai, S. Sakazawa, and S. Satoh, "Embedding watermarks into deep neural networks," in *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval*, pp. 269–277, June 2017.
- [24] B. D. Rouhani, H. Chen, and F. Koushanfar, "DeepSigns: a generic watermarking framework for ip protection of deep learning models," 2018, <https://arxiv.org/abs/1804.00750>.
- [25] S. Baluja, "Hiding images within images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 7, pp. 1685–1697, 2019.
- [26] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 126–137, ACM, Honolulu, HI, USA, July 2019.
- [27] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, IEEE, Venice, October 2017.
- [28] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, "Stargan: unified generative adversarial networks for multi-domain image-to-image translation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8789–8797, IEEE, Salt Lake City, UT, USA, June 2018.
- [29] H. Tang, H. Liu, and N. Sebe, "Unified generative adversarial networks for controllable image-to-image translation," *IEEE Transactions on Image Processing*, vol. 29, pp. 8916–8929, 2020.

- [30] F. Liang, L. Zhou, J. Zhong et al., "Evolutionary multitasking via explicit autoencoding," *IEEE Transactions on Cybernetics*, vol. 49, no. 9, pp. 3457–3470, 2018.
- [31] S. Liu, Q. Shi, and L. Zhang, "Few-shot hyperspectral image classification with unknown classes using multitask deep learning," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 59, no. 6, pp. 5085–5102, 2020.
- [32] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, Las Vegas, NV, USA, June 2016.
- [33] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, <https://arxiv.org/abs/1409.1556>.
- [34] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3730–3738, Santiago, Chile, December 2015.
- [35] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar, "Attribute and simile classifiers for face verification," in *Proceedings of the IEEE 12th international conference on computer vision*, pp. 365–372, IEEE, Kyoto, Japan, September 2009.
- [36] K. Chaudhari, "Pubfig. Website," 2021, <https://www.kaggle.com/datasets/kaustubhchaudhari/pubfig-dataset-256x256-jpg>.
- [37] M. Mozafari, "Bitmoji faces. Website," 2021, <https://www.kaggle.com/mostafamozafari/bitmoji-faces>.
- [38] D. P. Kingma and B. A. Jimmy, "Adam: a method for stochastic optimization," 2014, <https://arxiv.org/abs/1412.6980>.
- [39] J. Tao, S. Li, X. Zhang, and Z. Wang, "Towards robust image steganography," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 2, pp. 594–600, 2018.
- [40] J. Kim, M. Kim, H. Kang, and K. H. Lee, "U-gat-it: unsupervised generative attentional networks with adaptive layer-instance normalization for image-to-image translation," in *Proceedings of the International Conference on Learning Representations*, April 2019.
- [41] R. Chen, W. Huang, B. Huang, F. Sun, and B. Fang, "Reusing discriminators for encoding: towards unsupervised image-to-image translation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8168–8177, Seattle, WA, USA, June 2020.
- [42] M. Liu, Q. Li, Z. Qin, G. Zhang, P. Wan, and Z. Wen, "Blendgan: implicitly gan blending for arbitrary stylized face generation," *Advances in Neural Information Processing Systems*, vol. 34, 2021.
- [43] S. Wold, E. Kim, and G. Paul, "Principal component analysis," *Chemometrics and Intelligent Laboratory Systems*, vol. 2, no. 1-3, pp. 37–52, 1987.
- [44] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 11, 2008.
- [45] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.
- [46] Scikit, "Scikit-learn library. Website," 2021, <https://scikit-learn.org/stable/>.

Research Article

Integrating Temporal and Spatial Attention for Video Action Recognition

Yuanding Zhou,¹ Baopu Li,² Zhihui Wang ,¹ and Haojie Li¹

¹Dalian University of Technology, Dalian 116024, Liaoning Province, China

²Baidu Research, Sunnyvale, CA 94089, USA

Correspondence should be addressed to Zhihui Wang; zhwang@dlut.edu.cn

Received 14 February 2022; Revised 14 March 2022; Accepted 22 March 2022; Published 26 April 2022

Academic Editor: Beijing Chen

Copyright © 2022 Yuanding Zhou et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In recent years, deep convolutional neural networks (DCNN) have been widely used in the field of video action recognition. Attention mechanisms are also increasingly utilized in action recognition tasks. In this paper, we want to combine temporal and spatial attention for better video action recognition. Specifically, we learn a set of sparse attention by computing class response maps for finding the most informative region in a video frame. Each video frame is resampled with this information to form two new frames, one focusing on the most discriminative regions of the image and the other on the complementary regions of the image. After computing sparse attention all the newly generated video frames are rearranged in the order of the original video to form two new videos. These two videos are then fed into a CNN as new inputs to reinforce the learning of discriminative regions in the images (spatial attention). And the CNN we used is a network with a frame selection strategy that allows the network to focus on only some of the frames to complete the classification task (temporal attention). Finally, we combine the three video (original, discriminative, and complementary) classification results to get the final result together. Our experiments on the datasets UCF101 and HMDB51 show that our approach outperforms the best available methods.

1. Introduction

As an important communication medium, video contains a wealth of information. But this information used to be extracted and used manually, which is time-consuming and laborious. With the development of deep learning, attempts have been made to allow computers to extract information from videos. Many video-based deep learning tasks have emerged, such as video action localization [1], video captioning [2], and video question-answering [3]. The video action recognition task is to derive the behavior of a person in a video by analyzing the video content. This task is essentially a classification task where the input is a video and the output is action labels. This task has a wide range of application scenarios; most typically, it can detect violent action in surveillance videos and help police investigate and collect evidence [4].

With the development of deep learning, many excellent methods for video action recognition have emerged. Video is

composed of many frames, so the understanding of video should include the relationship between frames in addition to the image frames themselves. Therefore, the classical two-stream network [5] divided the video into two parts: spatial and temporal. Spatial part is the information of video frame, for which there are many excellent 2D CNN structures available, such as ResNet [6] and Inception [7], while the temporal information comes from the association between frames, and this part was obtained by optical flow. Finally the temporal and spatial information were integrated together to classify the video. Temporality is an important feature of video; many researchers spend their efforts on how to better capture the relationship between videos in the temporal dimension [8]. In addition to the temporal dimension, the information extraction of video frames itself is also an important part. So researchers also gradually put their efforts back to the images themselves in recent years. SlowFast [9] sampled the original video at different frame rates. The slow

path learned spatial information with few frames and the fast path learned temporal information with a large number of frames and then combined it with nonlocal network to model the relationship between frames from a global perspective. Video transformer [10] used transformer instead of convolution to compute the internal relationship of the whole video. But it was too computationally intensive to compute both temporal and spatial attention for each patch of each frame. So they proposed another architecture that temporal attention and spatial attention are separately applied one after the other. They found that the latter one not only reduced the computational effort significantly, but also had a higher accuracy in the end.

We find that previous attention methods tend to favor only one of temporal or spatial attention or treat all video frames with the same attention strategy, like the different frame rates of SlowFast [9], the transformer used by [10]. Inspired by previous approaches [11], we find that temporal and spatial attention can complement each other to improve the final classification. So in this paper, we first propose a spatial attention mechanism that extracts discriminative regions in video frames and resamples them into two new videos. These two videos are like a data augmentation of the original video. We then feed these two videos together with the original video into a temporal attention network with a frame selection strategy to filter out the most useful frames for classification task. Finally our network learns the most discriminative regions in these most useful frames, resulting in a more accurate result. At the same time, since the network where we extract spatial attention and the network that finally completes the classification task are the same, our extraction of discriminative regions in image frames is also getting accurate as well as the final classification accuracy. A positive beneficial cycle is formed to continuously improve our classification results.

The main innovations and contributions of this paper are as follows: (1) We propose a novel sparse attention mechanism for extracting important regions from video frames, and the method extracts discriminative regions while preserving contextual information. We leverage this proposed method as spatial attention. (2) We combine the spatial attention with our previously proposed frame selection strategy [12] to jointly form a novel network structure containing both temporal and spatial attention. (3) Experiments on two datasets commonly used for video action recognition, UCF101 and HMDB51, show that our approach outperforms the best available methods.

In Section 2, the structure of the proposed network, loss function, and other related contents will be introduced. In Section 3, the experimental results of our method and the implement details will be introduced. The advantages of our scheme will be summarized in Section 4.

2. Proposed Method

Inspired by [13], we find that the class peak responses typically correspond to strong visual cues residing inside regions of interest. As shown in Figure 1, we first feed the original video into a pretrained CNN with temporal attention (T-CNN) to

extract features (Features in Green). This part of features is sent to the spatial attention network to activate class response maps that allows the network to focus on the important part of the video frames. Based on the peaks of class response map each frame of the video is resampled into two new video frames. One of these two video frames focuses on the discriminative region of the image (the orange frame which enlarges the barbell part of the original frame) and the other focuses on the complementary part (the blue frame which enlarges the human body). These two branches then rearrange the video frames into two new videos in the same order as the original video. These two videos will also be fed into the T-CNN as new inputs. Each of these three branches is optimized by a cross-entropy loss function. Finally, the three video branches are jointly optimized to obtain a more accurate classification result.

2.1. Obtaining Class Peak Response Point. We first feed the video into T-CNN (which will be introduced in Section 2.3) that has been trained to extract the feature maps $\mathbf{X} \in \mathbb{R}^{T \times C \times H \times W}$, where T represents the number of frames, $H \times W$ is the size of the feature maps, and C is the number of channels. Then we feed the feature map into a global average pooling (GAP) layer and then go through a fully connected (FC) layer to get the classification score $\mathbf{x} \in \mathbb{R}^S$, where S is the number of categories in the dataset. We expand the feature maps along time dimension into T maps, each with dimension $\mathbf{Y} \in \mathbb{R}^{C \times H \times W}$. Then we let each of these maps go through a GAP and FC layer to get the classification score $\mathbf{y} \in \mathbb{R}^S$ for each frame. With the weight matrix of the FC layer $\mathbf{W}^{\text{fc}} \in \mathbb{R}^{C \times S}$, we can compute the class response map \mathbf{M}_s as

$$\mathbf{M}_s = \sum_{c=1}^C \mathbf{W}_{c,s}^{\text{fc}} \times \mathbf{Y}_c. \quad (1)$$

The class peak response of class c is defined as a local maximum of the corresponding class response map \mathbf{M}_c . The class peak point can be written as $P_c = \{(x_0, y_0), (x_1, y_1), \dots, (x_{N_s}, y_{N_s})\}$, where N_s is the number of valid peak points in the s -th class. We use these peak points to locate regions that are more important for the classification task and estimate a set of sparse attentions.

Experiments show that peaks in top-1 class response map tend not to cover all discriminative regions, while peaks in top-5 tend to contain the noise points. To seek a balance between these two methods of choosing peak points, we first calculate their entropy to determine the confidence of network predictions. If the confidence is high, we use peaks from the top-1 class response map, and if it is lower, we bring together the top-5 five class response maps to find the peak points. We denote the predicted probability of all S classes as $\mathbf{Prob} = \text{softmax}(\mathbf{y}) \in \mathbb{R}^S$ and use $\overline{\mathbf{Prob}} \in \mathbb{R}^S$ to denote the probability value of the top-5 classes. We compute the entropy as

$$H = - \sum_{i=1}^5 \mathbf{p}_i \log \mathbf{p}_i, \quad \mathbf{p}_i \in \overline{\mathbf{Prob}}. \quad (2)$$

We construct a response map \mathbf{R}_{map} with the following strategy:

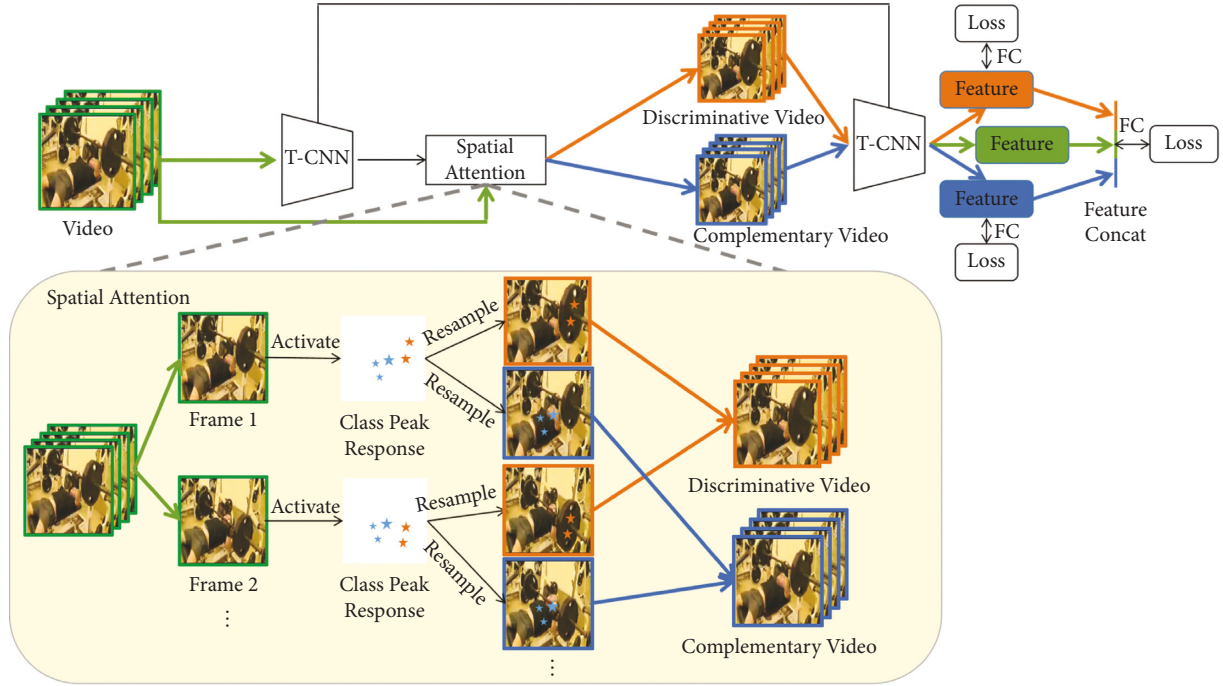


FIGURE 1: Network structure of our method.

$$\mathbf{R}_{\text{map}} = \begin{cases} \overline{\mathbf{M}}_1, & \text{if } H < \delta, \\ \sum_{k=1}^5 \overline{\mathbf{M}}_k, & \text{if } H > \delta, \end{cases} \quad (3)$$

where $\overline{\mathbf{M}} \in \mathbb{R}^{5 \times H \times W}$ is the class response maps corresponding to **Prob**. Then we use Min-Max Normalize to map the values of \mathbf{R}_{map} to $[0, 1]$.

$$\overline{\mathbf{R}}_{\text{map}} = \frac{\mathbf{R}_{\text{map}} - \min(\mathbf{R}_{\text{map}})}{\max(\mathbf{R}_{\text{map}}) - \min(\mathbf{R}_{\text{map}})}. \quad (4)$$

We denote their positions as $P = \{(x_1, y_1), (x_2, y_2), \dots, (x_{N_p}, y_{N_p})\}$, where N_p is the number of peaks we detected by the above procedure.

2.2. Computing Sparse Attention and Resampling. Reference [14] found that, in fine-grained image classification task, the obtained class peak points can be divided into two sets. One set is the discriminative region and the other is the complementary region, and learning these two sets separately is better than learning all class peak points together directly. Inspired by them, we preset a random number $\varphi_{(x,y)}$ from the uniform distribution between 0 and 1. We compare the response value $\overline{\mathbf{R}}_{\text{map}}$ of the peak point with this random number φ and group all points with response values greater than φ into one set P_{dis} and those less than into another set P_{com} .

$$\begin{cases} P_{\text{dis}} = \{(x, y) | (x, y) \in P \text{ if } \overline{\mathbf{R}}_{\text{map}(x,y)} \geq \varphi\}, \\ P_{\text{com}} = \{(x, y) | (x, y) \in P \text{ if } \overline{\mathbf{R}}_{\text{map}(x,y)} < \varphi\}. \end{cases} \quad (5)$$

The left part of Figure 2 is the original video frame, where the orange dot is the center point of the attention map (middle). As shown in Figure 2(a), points with high response values tend to correspond to discriminative regions, such as bow and arrow, and these peak points are generally grouped into the P_{dis} set. The points with low response values are usually localized at complementary regions as illustrated in Figure 2(b), that is, usually people in the video or the subject of the action, and these peak points will be grouped into the P_{com} set.

For each peak set, we compute a set of sparse attention $\mathbf{A} \in \mathbb{R}^{N_p \times H \times W}$ using Gaussian kernel.

$$\mathbf{A}_{i,x,y} = \begin{cases} R_{x_i,y_i} e^{-(x-x_i)^2+(y-y_i)^2/R_{x_i,y_i}\beta_1^2}, & \text{if } (x_i, y_i) \in P_{\text{dis}}, \\ \frac{1}{R_{x_i,y_i}} e^{-(x-x_i)^2+(y-y_i)^2/R_{x_i,y_i}\beta_2^2}, & \text{if } (x_i, y_i) \in P_{\text{com}}. \end{cases} \quad (6)$$

Both β_1 and β_2 are learnable parameters.

With the previously obtained sparse attention, we can resample the discriminative regions from the original video frames while also preserving the contextual information around the image regions. After the above series of operations, each video frame can be resampled to obtain two new frames, and we use \mathbf{Q}_{dis} to denote the feature map of the extracted discriminative branch and \mathbf{Q}_{com} to correspond to the feature map of the complementary branch.

$$\begin{cases} \mathbf{Q}_{\text{dis}} = \sum \mathbf{A}_i, & \text{if } (x_i, y_i) \in P_{\text{dis}}, \\ \mathbf{Q}_{\text{com}} = \sum \mathbf{A}_i, & \text{if } (x_i, y_i) \in P_{\text{com}}. \end{cases} \quad (7)$$

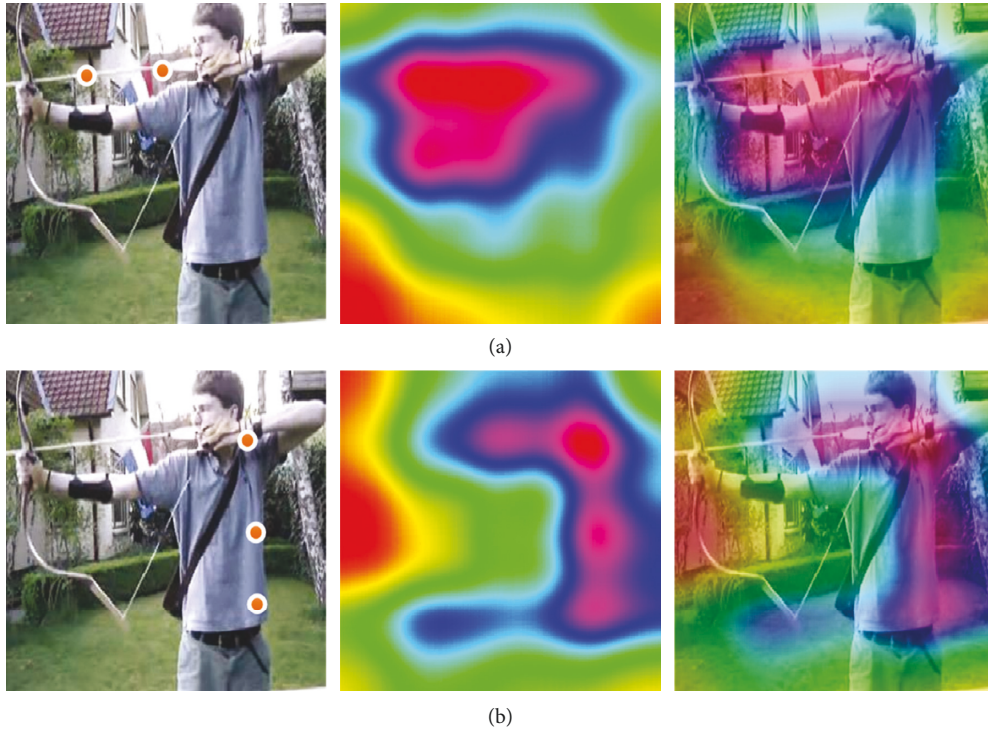


FIGURE 2: Visualization of discriminative and complementary branches: (a) discriminative branch; (b) complementary branch.

The resampling process is implemented using convolution following the method of [15] and can be embedded into the end-to-end training. So both β_1 and β_2 can be updated by the classification loss function. The input video has multiple frames and each frame of video can produce two frames according to the above method; we then line up all the discriminative and complementary branch images in the order of the original video input to form two new videos \mathbf{V}_{com} and \mathbf{V}_{dis} .

2.3. Network Structure and Loss Function. Essentially, the sparse attention we propose is for spatial attention of video images. Our input is an original video \mathbf{V}_o and the output is two videos \mathbf{V}_{com} and \mathbf{V}_{dis} . The video frames of \mathbf{V}_{dis} focus more on discriminative regions, while \mathbf{V}_{com} focuses on regions that are complementary.

As we mentioned before, in order to obtain the two new videos, we first feed the original video into a pretrained CNN with temporal attention (T-CNN) to extract features. Table 1 shows the network structure of T-CNN; “Dilation Conv(4)” means a dilation convolution with a dilation of 4 is used in the temporal dimension. T-CNN comes from a network structure that we obtained previously using neural architecture search [12]. In this work we explored how many frames are needed in each stage of the network. In fact, it is about allowing the network to focus on only the appropriate number of video frames to complete the final classification task. After we get the two new videos, \mathbf{V}_{com} and \mathbf{V}_{dis} , we will refeed them to T-CNN as new data to learn. These two new videos are equivalent to a data augmentation of our original input. So our method does not significantly improve the

number of model parameters, although the computational complexity increases.

Our loss function is a cross-entropy loss. Each input video will produce three predictions, which we denote as \mathbf{F}_o , \mathbf{F}_{com} , and \mathbf{F}_{dis} . These three predictions come from the original video \mathbf{V}_o , the discriminative video \mathbf{V}_{dis} , and the complementary video \mathbf{V}_{com} , respectively. Comparing them with the classification labels will produce three losses. We will also concatenate the features of the three videos together and pass them through a FC layer to obtain the fourth prediction $\mathbf{F}_{\text{total}}$. So our final loss function consists of four components, which can be written as

$$L(\mathbf{X}) = \sum_{i \in \{O, C, D\}} L_{cls}(\mathbf{F}_i, \mathbf{F}^*) + L_{cls}(\mathbf{F}_{\text{total}}, \mathbf{F}^*), \quad (8)$$

where L_{cls} denotes the cross-entropy loss and \mathbf{F}^* is the ground-truth label vector.

3. Experimental Results and Discussions

3.1. Datasets and Implementation Details. To evaluate the effectiveness of our proposed method, we have done experiments on two common datasets for video action recognition, UCF101 and HMDB51.

The UCF101 dataset [16] has 13,320 videos from 101 action categories. Each of these categories can be divided into 25 groups, each containing 4–7 action videos. This dataset is highly diverse in terms of motion and varies greatly in terms of camera movement, object appearance and pose, object scale, point of view, cluttered backgrounds, lighting conditions, etc.

TABLE 1: Network structure of T-CNN.

Input: $3 \times 16 \times 224 \times 224$	
Stage 1	Conv 3–32 + BN + ReLU
	Conv 32–32 + BN + ReLU
Stage 2	Conv 32–64 + BN + ReLU
	Conv 64–64 + BN + ReLU
Stage 3	Conv 64–96 + BN + ReLU
	Conv 96–96 + BN + ReLU
Stage 4	Conv 96–160 + BN + ReLU
	Conv 160–160 + BN + ReLU
	Conv 160–160 + BN + ReLU
	Dilation Conv(4) 160–160 + BN + ReLU
	Conv 160–160 + BN + ReLU
	Dilation Conv(4) 160–160 + BN + ReLU
	Conv 160–160 + BN + ReLU
	Dilation Conv(4) 160–160 + BN + ReLU
	Conv 160–160 + BN + ReLU
	Conv 160–160 + BN + ReLU
	Conv 160–160 + BN + ReLU
Stage 5	Dilation Conv(4) 160–160 + BN + ReLU
	Conv 160–224 + BN + ReLU
	Conv 224–224 + BN + ReLU
	Conv 224–224 + BN + ReLU
	Dilation Conv(2) 224–224 + BN + ReLU
	Conv 224–224 + BN + ReLU
Stage 6	Dilation Conv(4) 224–224 + BN + ReLU
	Conv 224–288 + BN + ReLU
	Conv 288–288 + BN + ReLU
	Conv 288–288 + BN + ReLU
	Dilation Conv(2) 288–288 + BN + ReLU
	Conv 288–288 + BN + ReLU
	Dilation Conv(2) 288–288 + BN + ReLU
	Conv 288–288 + BN + ReLU
Stage 7	Conv 288–512 + BN + ReLU
	Conv 512–512 + BN + ReLU
	Conv 512–512 + BN + ReLU
	Dilation Conv(2) 512–512 + BN + ReLU
	Global average pooling
Fully connected layer	
Softmax	
Classification result	

The HMDB51 dataset [17] contains 51 action categories, a total of 6849 videos, and each action contains at least 51 videos. The action categories can be divided into four major categories: (1) general facial actions (laughing, chewing); (2) facial and object actions (smoking, eating); (3) human body actions (hugging, inversion); (4) interactive actions with objects (horse riding, archery).

For both video datasets, during training, we sample 16 consecutive frames from each video, and each frame is converted to 256×342 resolution by preprocessing. And then we randomly crop 224×224 pixels from the frame and

feed them into the network. To make a fair comparison with other methods, we follow the common reference method [18]. We divide each video into 10 clips equally, with each clip including 16 video frames, resize the short edge of each image to 224 pixels, and cut three 224×224 crops from the left, middle, and right of the image. Each crop of each clip is called a “view,” so we have 30 views, and the final prediction result of each video is obtained by averaging the softmax scores of these 30 views.

The whole model is trained for 150 epochs with a batch size of 16. We use SGD optimizer with 0.9 momentum and 4×10^{-5} weight decay. The learning rate strategy uses cosine annealing learning rate schedule [19]. The initial learning rate was 0.1 and the lowest was 1×10^{-4} . The dropout probability is 0.5 after the final GAP layer. Finally, it is sent to the linear layer to classify according to the number of classes of each dataset.

3.2. Comparison with SOTA. On the two commonly used datasets, UCF101 and HMDB51, we compare the proposed method with the SOTA methods. Since our method uses only RGB images as input, when comparing with other methods that have multiple input modalities like I3D, we only compare with their results obtained with RGB modality. From the results of the comparisons in Tables 2 and 3, we can see that the classification accuracy of our method on both datasets exceeds the best available methods.

The advantage of our method comes first from our spatial attention. From the results, the network of T-CNN with only temporal attention has lower accuracy than TSM and I3D RGB on both datasets. In particular, for I3D RGB, T-CNN is 0.3% lower than it on UCF101 and 1.5% lower than it on HMDB51. When the spatial attention proposed in this paper is added, our method achieves a reversal on both datasets. This effect is related to these two datasets, which are more sensitive to spatial information, so the increase of attention to spatial information will produce such a huge improvement (1.4% for UCF101 and 1.9% for HMDB51). Then there is the fact that the spatial attention in this paper is finally externalized to two new data inputs, which actually has the effect of data augmentation. This is very important because both UCF101 and HMDB51 are easy to overfit. Data augmentation helps to improve generalization ability and reduce the occurrence of overfitting.

The second advantage of our approach comes from the fact that we integrate spatial and temporal attention, allowing them to complement each other and improve the final classification accuracy. Not all video frames have positive implications for classification. As shown in Figure 3, this image is difficult to classify based on the original picture and spatial attention. It is easy to be classified as “holding something” rather than “shooting an arrow.” At this point, we can rely on temporal attention in the frame selection strategy to reduce our chances of picking this image frame, thus reducing the number of misleading cases.

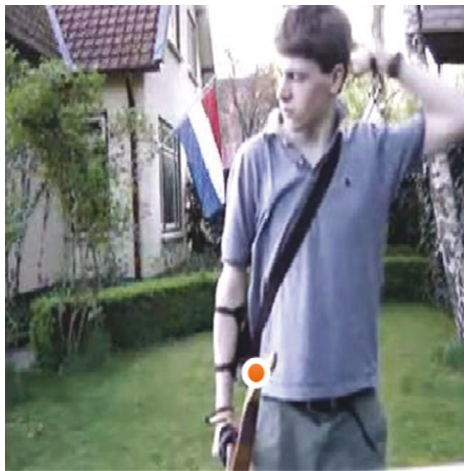
There are also shortcomings in our method. From the last column in Tables 2 and 3, we can see that the computational complexity of our method has increased several

TABLE 2: Comparisons with other methods on UCF101 dataset.

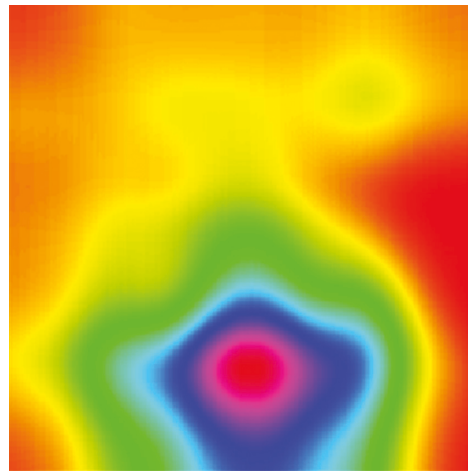
Model	Pretraining dataset	Accuracy (%)	GFLOPs
C3D [20]	Sports-1M	82.3	38.57
TRN [21]	—	83.5	83.83
Res3D [22]	Sports-1M	85.8	—
P3D [23]	Imagenet + Sports-1M	88.6	18.51
T3D [24]	Kinetics-400	90.3	—
TSN [8]	Imagenet + Kinetics-400	91.1	80
R(2 + 1)D [25]	Sports-1M	93.6	41.69
TSM [26]	Kinetics-400	95.5	32.88
I3D RGB [27]	Imagenet + Kinetics-400	95.6	108
T-CNN [12]	Kinetics-400	95.3	15.78
T-CNN + spatial	Kinetics-400	96.7	52.3

TABLE 3: Comparisons with other methods on HMDB51 dataset.

Model	Pretraining dataset	Accuracy (%)	GFLOPs
Res3D [22]	Sports-1M	54.9	—
T3D [24]	Kinetics-400	59.2	—
R(2 + 1)D [25]	Sports-1M	66.6	41.69
TSM [26]	Kinetics-400	73.6	32.88
I3D RGB [27]	Imagenet + Kinetics-400	74.8	108
T-CNN [12]	Kinetics-400	73.3	15.78
T-CNN + spatial	Kinetics-400	75.2	52.3



(a)



(b)

FIGURE 3: An example of classification error based only on spatial attention. (a) original frame. (b) Spatial attention.

times compared to T-CNN. The T-CNN has the lowest computational complexity among existing methods (15.78 GFLOPs), but with the addition of the spatial attention part, the computational complexity comes directly to the back half of the list. This is mainly due to the fact that our spatial attention resamples two new videos into the network, which equates to one video input that needs to be computed 3 times through the network, plus the fact that we need to compute the class response maps and sparse attentions for each frame and resample them. All of them add to the computational complexity.

3.3. Effects of Different Extraction Branches. To verify the effects of each branch, we tried to omit one or more branches and observe their effects on the final classification results. From Table 4, we can draw the following conclusions. (1) Both O + D mode and O + C mode are improved for the final classification accuracy. It indicates that both complement and discriminative regions are helpful for classification, and it also verifies that the spatial attention extraction method in this paper is effective. (2) In the absence of the complementary branch, our overall accuracy decreases the least (from 96.7 to 96.3), indicating that the complementary

TABLE 4: Ablation study on UCF1051 dataset based on different branches.

Branches	Original branch	Discriminative branch	Complement branch	Total accuracy
O	95.3	—	—	95.3
O + D	96.0	95.8	—	96.3
O + C	95.5	—	95.0	95.8
C + D	—	95.9	95.0	96.2
O + C + D	96.2	95.6	94.8	96.7

branch is indeed the region containing the least discriminative information compared to the other branches. However, the accuracy of the classification still decreases when this part is missing, suggesting that sometimes the subject of the action can also play a crucial role in the classification task.

4. Conclusions

In this paper we integrate temporal and spatial attention to construct a network structure. We learn a set of sparse attention by computing class response maps. It selectively collects visual evidence of dynamic information areas based on image content and surrounding context. Based on these regions obtained by spatial attention we resampled two new videos. These new videos are fed into the network as completely new data, enhancing the generalization ability of our network structure. We then feed these two videos with the spatial attention together with the original video into a temporal attention network. So our network learns the most discriminative regions in these most useful frames, resulting in a more accurate result. And the network where we extract spatial attention is the same as the network that finally completes the classification task. So our extraction of discriminative regions in image frames is also getting accurate as well as the final classification accuracy. A positive cycle is formed to continuously improve the classification results. Integrating attention in temporal and spatial is actually consistent with human vision. We also recognize the other person's action by observing key object information in consecutive actions. Extensive experimental results on some benchmark datasets illustrate the promising performance of the proposed scheme.

Data Availability

The datasets used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

- [1] F. Wang, G. Wang, Y. Du, Z. He, and Y. Jiang, "A two-stage temporal proposal network for precise action localization in untrimmed video," *International Journal of Machine Learning and Cybernetics*, vol. 12, no. 8, pp. 2199–2211, 2021.
- [2] P. Li, P. Zhang, and X. Xu, "Graph convolutional network meta-learning with multi-granularity POS guidance for video captioning," *Neurocomputing*, vol. 472, pp. 294–305, 2022.
- [3] J. Zhang, J. Shao, R. Cao, L. Gao, X. Xu, and H. T. Shen, "Action-centric relation transformer network for video question answering," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 1, pp. 63–74, 2022.
- [4] M. Bruno, B. Lavi, Z. Dias, and R. Anderson, "Harnessing high-level concepts, visual, and auditory features for violence detection in videos," *Journal of Visual Communication and Image Representation*, vol. 78, Article ID 103174, 2021.
- [5] K. Simonyan and A. Zisserman, "Two-Stream convolutional networks for action recognition in videos," in *Proceedings of the 27th International Conference on Neural Information Processing Systems NIPS*, pp. 568–576, Montreal Canada, December 2014.
- [6] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition CVPR*, pp. 770–778, Las Vegas, NV, USA, June 2016.
- [7] C. Szegedy, W. Liu, Y. Jia et al., "Going deeper with convolutions," in *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition CVPR*, pp. 1–9, Boston, MA, June 2015.
- [8] L. Wang, Y. Xiong, Z. Wang et al., "Temporal segment networks: towards good practices for deep action recognition," in *Proceedings of the European Conference on Computer Vision ECCV*, no. 8, pp. 20–36, Amsterdam, The Netherlands, October 2016.
- [9] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "SlowFast networks for video recognition," in *Proceedings of the 2019 International Conference on Computer Vision ICCV*, pp. 6201–6210, Seoul, Korea, November 2019.
- [10] G. Bertasius, H. Wang, and L. Torresani, "Is space-time attention all you need for video understanding?" in *Proceedings of the 38th International Conference on Machine Learning ICML*, pp. 813–824, July 2021.
- [11] Y. Wang, Z. Chen, H. Jiang, S. Song, Y. Han, and G. Huang, "Adaptive focus for efficient video recognition," in *Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision*, Montreal, QC, Canada, October 2021.
- [12] Y. Zhou, B. Li, Z. Wang, and H. Li, "Video action recognition with neural architecture search," in *Proceedings of the 13th Asian Conference on Machine Learning ACML*, pp. 1675–1690, November 2021.
- [13] Y. Zhou, Y. Zhu, Q. Ye, Q. Qiu, and J. Jiao, "Weakly supervised instance segmentation using class peak response," in *Proceedings of the CVPR*, pp. 3791–3800, Salt Lake City, Utah, June 2018.
- [14] Y. Ding, Y. Zhou, Y. Zhu, Q. Ye, and J. Jiao, "Selective sparse sampling for fine-grained image recognition," in *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision ICCV*, pp. 6598–6607, Seoul, Korea (South), November 2019.
- [15] A. Recasens, P. Kellnhofer, S. Simon, W. Matusik, and A. Torralba, "Learning to zoom: a saliency-based sampling layer for neural networks," in *Proceedings of the ECCV*, no. 9, pp. 52–67, Munich, Germany, September 2018.

- [16] K. Soomro, Amir Roshan Zamir, and M. Shah, "UCF101: a dataset of 101 human actions classes from videos in the wild," 2012, <https://arxiv.org/abs/1212.0402>.
- [17] H. Kuehne, H. Jhuang, E. Garrote, T. A. Poggio, and T. Serre, "HMDB: a large video database for human motion recognition," in *Proceedings of the 2011 International Conference on Computer Vision ICCV*, pp. 2556–2563, Barcelona, Spain, November 2011.
- [18] Y. Li, B. Ji, X. Shi, J. Zhang, B. Kang, and L. Wang, "TEA: temporal excitation and aggregation for action recognition," in *Proceedings of the CVPR*, pp. 906–915, Seattle, Washington, June 2020.
- [19] I. Loshchilov and F. Hutter, "SGDR: stochastic gradient descent with warm restarts," in *Proceedings of the ICLR*, Palais des Congrès Neptune, Toulon, Fr, April 2017.
- [20] Du Tran, L. D. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proceedings of the 2015 IEEE International Conference on Computer Vision ICCV*, pp. 4489–4497, Santiago, Chile, 2015.
- [21] B. Zhou, A. Andonian, A. Oliva, and A. Torralba, "Temporal relational reasoning in videos," in *Proceedings of the European Conference on Computer Vision ECCV*, no. 1, pp. 831–846, Munich, Germany, September 2018.
- [22] D. Tran, J. Ray, S. Zheng, S. F. Chang, and M. Paluri, "ConvNet architecture search for spatiotemporal feature learning," 2017, <https://arxiv.org/abs/1708.05038>.
- [23] J. Wei, H. Wang, Y. Yi, Q. Li, and D. Huang, "P3D-Ctn: Pseudo-3D convolutional tube network for spatio-temporal action detection in videos," in *Proceedings of the 2019 IEEE International Conference on Image Processing ICIP*, pp. 300–304, Taipei, Taiwan, September 2019.
- [24] D. Ali, M. Fayyaz, V. Sharma et al., "Temporal 3D ConvNets: new architecture and transfer learning for video classification," 2017, <https://arxiv.org/abs/1711.08200>.
- [25] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, "A closer look at spatiotemporal convolutions for action recognition," in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition CVPR*, pp. 6450–6459, Salt Lake City, UT, USA, June 2018.
- [26] J. Lin, C. Gan, and S. Han, "TSM: Temporal shift module for efficient video understanding," in *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision ICCV*, pp. 7082–7092, 2019.
- [27] J. Carreira, A. Zisserman, and Q. Vadis, "Action recognition? A new model and the kinetics dataset," in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition CVPR*, pp. 4724–4733, Honolulu HI, USA, July 2017.

Research Article

Separable Reversible Data Hiding in Encrypted VQ-Encoded Images

Fang Cao ^{1,2}, Yujie Fu,³ Heng Yao ³, Mian Zou ⁴, Jian Li ⁵ and Chuan Qin ³

¹College of Information Engineering, Shanghai Maritime University, Shanghai 200135, China

²Guangxi Key Lab of Multi-source Information Mining & Security, Guangxi Normal University, Guilin 541004, China

³School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China

⁴School of Mechanical Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China

⁵School of Cyber Security, Qilu University of Technology (Shandong Academy of Sciences), Shandong Provincial Key Laboratory of Computer Networks, Jinan 250353, China

Correspondence should be addressed to Jian Li; ljian20@gmail.com

Received 21 December 2021; Accepted 16 February 2022; Published 23 April 2022

Academic Editor: Beijing Chen

Copyright © 2022 Fang Cao et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In this paper, a reversible data-hiding scheme in encrypted, vector quantization (VQ) encoded images is proposed. During image encryption, VQ-encoded image, including codebook and index table, is encrypted by content owner with stream-cipher and permutation to protect the privacy of image contents. As for additional-data embedding, a baseline method is first proposed and its corresponding optimized method is then given. By grouping one high-occurrence index with one or multiple low-occurrence indices, a series of index groups are constructed. Thus, by modifying the high-occurrence index to the corresponding index within the same group according to the current to-be-embedded bits, data embedding can be realized. The optimal hiding capacity is obtained by optimizing the coefficient vector for different types of index groups. Separable operations of data extraction, image decryption, and recovery can be achieved on the receiver side based on the availability of the encryption and data-hiding keys. Experimental results show that our scheme can achieve high hiding capacity and satisfactory directly decrypted image quality and guarantee security and reversibility simultaneously.

1. Introduction

With the rapid development of digital communication and signal processing, a large amount of multimedia data, such as image, video, and audio, are transmitted on networks. However, secure management for multimedia data with privacy protection is an inevitable issue and also is one meaningful research topic. Reversible data-hiding (RDH) is an emerging technique which has greatly attracted researchers' interests in recent years [1–4]. As for the RDH technique, data hider can embed additional data into cover image reversibly, which means the original cover image can be completely recovered after extracting the embedded data. Some representative RDH schemes, such as difference expansion (DE) [1], histogram shifting (HS) [2], and

prediction-error expansion (PEE) [3], have been proposed in the past few years. In addition to the RDH schemes for gray scale images, there are a lot of RDH schemes developed for color images [5], compressed images [6], and halftone images [7].

Vector quantization (VQ) is an effective image encoding method, which can be utilized for image compression [8]. During the process of VQ encoding, the original uncompressed image I_o is first divided into a series of nonoverlapping blocks. For each image block, Euclidean distances between the block with the L code words in the trained codebook C are calculated, and the index of the code word with the minimum Euclidean distance is recorded into the index table T as the encoded result for the current block. During VQ decoding, according to the indices in the index

table **T**, all image blocks can be easily decoded as the corresponding code words in the codebook **C** to form the VQ-decoded image **I**. Figure 1 illustrates an example of VQ-encoded image, in which a gray scale image with the size of $M \times N$ is compressed to an index table sized $M/n \times N/n$, where $n \times n$ is the size of divided blocks. Each value in the index table, corresponding to one $n \times n$ block, can be represented with $\log_2 L$ bits. Thus, for the whole image, the compression ratio can be calculated as $8 \times n^2 / \log_2 L$. Generally speaking, a codebook with more code words, i.e., larger L , can lead to better visual quality of VQ-decoded image.

In recent years, a number of RDH schemes have been developed for VQ-encoded images in the plaintext form [9–16]. Chang et al. proposed a RDH scheme in VQ-encoded images based on a de-clustering strategy [9], in which two de-clustering methods were used with the minimum-spanning-tree and a short-spanning-path. Lee et al. modified VQ-encoded images by the side-matched VQ (SMVQ) technique to form a transformed image, and exploited the distribution of this transformed image to achieve high hiding capacity and low bit rate [10]. Kieu and Ramroach utilized the joint neighboring coding method to realize reversible steganographic scheme for VQ indices [12], in which the differences between the current index; the left, upper, and top-left neighboring indices; and their combinations were used to hide additional bits. In Ref. [15], two RDH schemes for VQ-encoded images were proposed based on switching-tree coding and dynamic-tree coding. These two schemes performed data embedding by choosing one of the possible index encoding ways when multiple ways were available to encode the index, and the outputted codes can be decoded to original VQ index table with the conventional decoder. Pan and Wang proposed a RDH scheme for two-stage VQ-encoded image based on search-order coding (SOC) in Ref. [16]. SOC can employ the correlation of indices to obtain better compression ratio, thus, the combination of SOC and data hiding in this scheme can achieve both high performances for compression ratio and hiding capacity.

Due to the current prosperity of cloud storing and computing, a vast amount of personal data are stored and processed on the cloud to alleviate computation burden on user clients [17, 18]. But, in order to protect user privacy, it is better to first encrypt user data before uploading onto cloud. Thereby, for the convenience of data management and retrieval, RDH in encrypted images (RDHEI) has attracted extensive interest in the field of multimedia security. According to when the space for accommodating additional data was created, i.e., before or after image encryption, embedding mechanisms of most RDHEI schemes can be categorized into two types: vacating room after encryption (VRAE) [19–28] and reserving room before encryption (RRBE) [29–34]. In addition, some researchers introduced homomorphic encryption (HE) into RDHEI [35–38], which can realize the operations of data embedding directly in encrypted domain. A brief review of the related works on RDHEI is given in Section 2.

In this work, we focus on RDH in encrypted, VQ-encoded image. An encryption method for VQ-encoded image is first designed for the codebook and the index table, respectively. Before conducting additional-data embedding in the encrypted index table, all VQ indices are sorted according to their occurrence numbers. A baseline method of data embedding is proposed based on constructing index groups for one high-occurrence index and one low-occurrence index each time, and then we improve the baseline method through generalized index grouping for multiple low-occurrence indices. By modifying the high-occurrence index to the corresponding index within the same group according to the current to-be-embedded bits, additional-data embedding can be achieved, and the optimal hiding capacity is obtained by optimizing coefficient vector for different types of index groups. Separable operations of data extraction, image decryption, and recovery can be realized on the receiver side based on the availability of the encryption and data-hiding keys. The proposed scheme can achieve satisfactory performances of hiding capacity and directly decrypted image quality and guarantee security and reversibility simultaneously.

The remaining parts of the paper are organized as follows. Section 2 gives a brief review of related works about RDHEI. Section 3 introduces the baseline of the proposed scheme, including image encryption, additional-data embedding, data extraction, and image recovery. Section 4 gives performance optimization for additional-data embedding procedure of the baseline method in Section 3, which consists of generalized index grouping, multiple-bits embedding, and hiding capacity optimization. Section 5 presents experimental results and analysis. Conclusions are drawn in Section 6.

2. Related Works

An effective RDHEI framework can be described as: the content-owner encrypts the original image with encryption key and then sends the encrypted image to the data hider; the data hider embeds additional data into the encrypted image with data-hiding key to produce the marked, encrypted image; and the authorized receiver implements data extraction, image decryption, and image recovery on the marked, encrypted image according to encryption key and data-hiding key. In the following, three main categories of RDHEI schemes are briefly reviewed.

2.1. VRAE-Based Schemes. In Ref. [19], the encrypted image with stream cipher was segmented into a number of non-overlapping blocks, and by flipping the three LSBs of different parts of pixels, one bit of additional data can be embedded into each block. The receiver can achieve data extraction and image recovery through estimation with a fluctuation function. Hong et al. improved the order of data extraction and block recovery and introduced a side-match strategy to increase the accuracy of the extracted data and recovered image [20]. Liao and Shu utilized the absolute mean difference of neighboring pixels to measure the

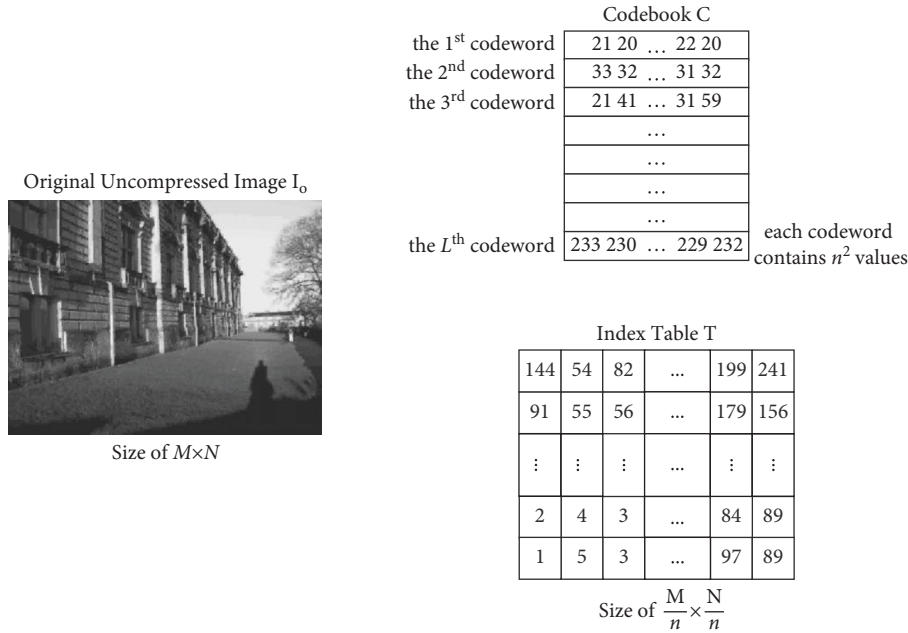


FIGURE 1: An example of VQ-encoded image.

recovery accuracy of image blocks after decryption [21]. Different from Refs. [19–21] that data extraction must be conducted after image decryption, a separable RDHEI scheme was proposed in Ref. [22], which means that the operations of image decryption and data extraction can be realized on the receiver side independently. A public key modulation mechanism was employed in Ref. [23] to achieve data embedding without accessing the secret encryption key. In addition, a powerful two-class SVM classifier was presented to differentiate the encrypted and nonencrypted patches, leading to recovering the embedded data and original image correctly. Huang et al. proposed to encrypt the original image in a blockwise manner [24], which can retain the correlation within the pixels of each encrypted block. Then, traditional RDH methods in plaintext images can be used in encrypted image for data hiding. In Ref. [25], a RDHEI scheme with an adaptive encoding strategy was presented, which adaptively compressed the MSB layers of embeddable blocks according to occurrence frequency of MSB and then embedded additional data together with reversed Huffman codewords and auxiliary data. Yi and Zhou first presented a parametric binary tree labeling (PBTL) algorithm to label pixels in two different types, and then, they proposed a PBTL-RDHEI scheme in encrypted images, which can achieve data embedding by pixel labeling and bit replacement effectively [28].

2.2. RRBE-Based Schemes. In order to avoid the errors on data extraction or image recovery, Ma et al. proposed a scheme by reserving room before encryption with a traditional RDH method in Ref. [29], which can acquire the complete reversibility. In Ref. [30], some pixels in the original plaintext image were first predicted before encryption, thus, additional data can then be embedded in the

prediction errors. A benchmark encryption algorithm was applied to the rest pixels and a specific encryption algorithm was designed to encrypt prediction errors. Cao et al. considered that an image patch can be linearly represented by some atoms in an over-complete dictionary through sparse coding [31], and the residual errors can be encoded and self-embedded in the original image. Thereby, a large extra room can be created before image encryption, and the data hider can embed more additional data into the encrypted image based on this strategy of patch-level sparse representation. Puteaux et al. proposed a new reversible method with most significant bit (MSB) prediction [32], which can achieve a high hiding capacity. During the preprocessing, a location map was produced by detecting prediction errors. Through MSB substitution, additional data can be embedded and the embedding rate was close to 1 bpp. Yin et al. proposed a RDHEI scheme based on multi-MSB prediction and Huffman coding [33]. Before image encryption with a stream cipher, multi-MSB of each pixel was predicted and marked with Huffman coding in the original image as the preprocessing. Thus, additional data can be embedded into the encrypted image through multi-MSB substitution.

2.3. HE-Based Schemes. Chen et al. proposed a RDH scheme for encrypted signals based on Paillier public key encryption, and applied it on digital images [35], in which each pixel value was divided into two parts encrypted, respectively. Then, two encrypted LSBs of each pixel pair were modified to hide one bit with the help of homomorphism. In Wu et al.’s scheme [36], each unit in the original image was segmented into three components with energy transfer equation, and each component was encrypted by Paillier homomorphic encryption. The data hider can embed additional bits into the encrypted image by using the properties of Paillier

homomorphism. A separable RDHEI scheme based on additive homomorphism and pixel value ordering (PVO) was given in Ref. [37]. Additive homomorphism applied in this scheme can guarantee that the performance of embedding rate for PVO in an encrypted domain can approximate to that in plaintext domain without involving data expansion. In Ref. [38], Xiang and Luo proposed to form mirroring ciphertext groups (MCGs) by replacing encrypted host pixels with encrypted reference pixels in the same group. In an MCG, the reference ciphertext pixel remained unchanged as a reference while the data hider can embed additional data into the LSBs of host encrypted pixels with homomorphic multiplication.

The abovementioned RDHEI schemes mainly focused on the encrypted, uncompressed gray scale image. In addition, some schemes have also been designed for other kinds of cover data, such as JPEG-encoded image [39–41], palette image [42], 2D vector graphic [43], and 3D mesh model [44], in the encrypted domain. However, to the best of our knowledge, there are few reported works about RDHEI of VQ-encoded images currently.

3. Baseline of the Proposed Scheme

Figure 2 presents the framework of the proposed scheme for RDH in encrypted VQ-encoded images. As shown in Figure 2(a), on the content-owner side, with encryption key $K_e = \{K_e^{(1)}, K_e^{(2)}\}$, encryption for VQ-encoded image can be divided into two steps: codebook encryption and index table encryption, respectively. Then, after receiving the encrypted, VQ-encoded image, through index grouping and data-hiding key K_h , additional data can be embedded on the data-hider side, see Figure 2(b). On the receiver side, we can extract additional data and restore the VQ-encoded image. It can be seen from Figure 2(c) that additional data can be extracted with data-hiding key K_h ; receiver can obtain a decrypted image which is similar to the original image with encryption key K_e ; when the receiver has both encryption key K_e and data-hiding key K_h , the embedded data can be successfully extracted and the VQ-encoded image can also be perfectly recovered. Details of our baseline scheme are introduced as follows.

3.1. VQ-Encoded Image Encryption. As we know, a VQ-encoded image consists of one codebook \mathbf{C} and an index table \mathbf{T} . Hence, in order to guarantee the security, VQ-encoded image encryption can be divided into two parts, i.e., codebook encryption and index table encryption.

Suppose VQ codebook \mathbf{C} contains L code words, and in each code word, there are n^2 decimal values. Denote $P_{i,j}$ as the j^{th} value of the i^{th} code word in the codebook \mathbf{C} , where $i = 1, 2, \dots, L, j = 1, 2, \dots, n^2$, and the value of $P_{i,j}$ can be represented as eight binary bits:

$$P_{i,j,k} = \left\lfloor \frac{P_{i,j}}{2^{k-1}} \right\rfloor \bmod 2, \quad k = 1, 2, \dots, 8, \quad (1)$$

where $P_{i,j,k}$ denotes the k^{th} bit of $P_{i,j}$. A sequence of pseudo-random bits $S_{i,j,k}$ ($i = 1, 2, \dots, L, j = 1, 2, \dots, n^2, k = 1, 2, \dots, 8$)

is generated with encryption key $K_e^{(1)}$. The operation of bitwise exclusive-or (XOR) is performed on all L code words for codebook encryption:

$$\begin{aligned} P_{i,j,k}^{(e)} &= P_{i,j,k} \oplus S_{i,j,k}, \\ P_{i,j}^{(e)} &= \sum_{k=1}^8 P_{i,j,k}^{(e)} \cdot 2^{k-1}, \end{aligned} \quad (2)$$

where $P_{i,j}^{(e)}$ denotes the j^{th} encrypted value in the i^{th} code word after stream-cipher encryption. After all code words in the codebook \mathbf{C} are encrypted, the encrypted codebook \mathbf{C}_e is obtained.

As for the index table \mathbf{T} sized $M/n \times N/n$, all the index values in \mathbf{T} are permuted with the encryption key $K_e^{(2)}$.

$$\mathbf{T}_e = \text{perm}(\mathbf{T}, K_e^{(2)}), \quad (3)$$

where $\text{perm}(\cdot)$ denotes the permutation function, and \mathbf{T}_e is the encrypted index table. The security can be guaranteed because the codebook \mathbf{C} is encrypted by the stream cipher while permuting the index table \mathbf{T} . The key space of index table permutation can be calculated as $(M/n \times N/n)!$. As for an original uncompressed image sized 512×512 ($M = N = 512$), when block size is chosen as 4×4 ($n = 4$), the whole key space of index table permutation is: $[(512/4) \times (512/4)]! = 16384!$. The codebook encryption based on stream cipher can be considered to further strengthen the security of encryption, even when the permutation key $K_e^{(2)}$ is leaked or cracked. After the VQ-encoded image encryption for codebook and index table, \mathbf{C}_e and \mathbf{T}_e are transmitted to the data-hider side together for additional-data embedding.

3.2. Additional-Data Embedding. In our scheme, after receiving \mathbf{C}_e and \mathbf{T}_e , data hider first counts the occurrence numbers of VQ indices in the encrypted index table \mathbf{T}_e . Initially, the occurrence numbers of indices corresponding to all L code words in the encrypted codebook \mathbf{C}_e are set as zero. When a VQ index is scanned in \mathbf{T}_e , its occurrence number is increased by one. That is to say, for the VQ index k , its occurrence number γ_k ($k = 1, 2, \dots, L$) can be calculated as:

$$\gamma_k = \sum_{x=1}^{M/n} \sum_{y=1}^{N/n} \phi(k, T_{x,y}), \quad (4)$$

where $T_{x,y}$ denotes the index value at coordinate (x, y) in the index table \mathbf{T}_e , and $\phi(\cdot)$ is a counting function returning 1 or 0. When the current VQ index $T_{x,y}$ is equal to k , ϕ returns 1; otherwise, ϕ returns 0. Generally, occurrence numbers of VQ indices in the index table are not uniform for a natural image. Figure 3 shows the distribution of occurrence numbers of VQ indices ($L = 128$) for image *Lena*, in which X axis and Y axis denote the index values and their corresponding occurrence numbers, respectively. We can observe from Figure 3 that some VQ indices occur frequently while some VQ indices are not used at all.

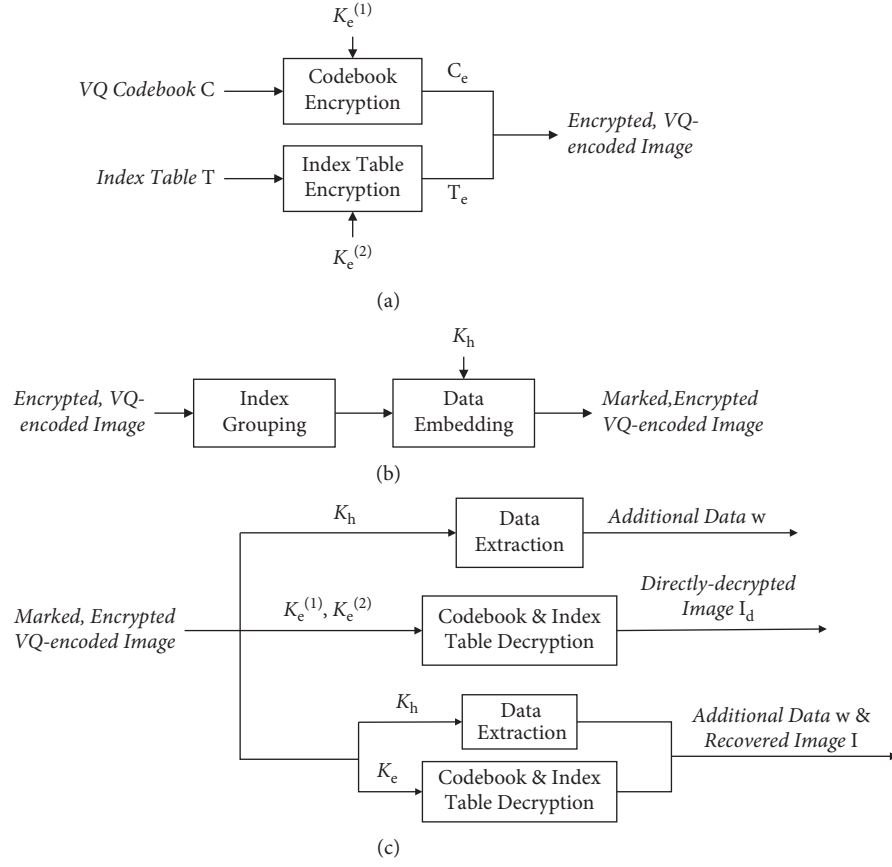


FIGURE 2: Framework of the proposed scheme. (a) VQ-encoded image encryption, (b) additional-data embedding, (c) data extraction and image recovery.

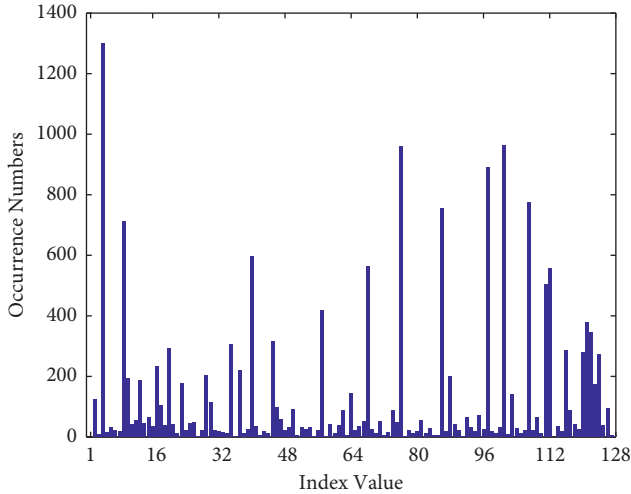


FIGURE 3: Histogram of VQ index values for image *Lena* ($L = 128$).

After scanning the whole index table T_e , the L different kinds of VQ indices are sorted according to their corresponding occurrence numbers γ_k ($k = 1, 2, \dots, L$) in the descending order. In the proposed scheme, VQ indices with higher and lower occurrence numbers are utilized to achieve additional-data embedding. Note that the distribution of VQ indices is not changed before and after VQ-encoded image

encryption in our scheme. The procedure of additional-data embedding includes two stages: (1) index grouping and (2) data embedding, which are described in detail as follows.

3.2.1. A Index Grouping. Denote the sorted L VQ indices as c_1, c_2, \dots, c_L , and their corresponding occurrence numbers are $\gamma_1, \gamma_2, \dots, \gamma_L$. We define the index set $\{c_\alpha, c_{\alpha+1}, \dots, c_L\}$ with lower occurrence numbers as Φ , where α is a threshold satisfying $\gamma_\alpha \leq \sigma$, and σ is a pre-determined parameter, which is discussed in Section 5. The relationship between α and σ can be represented as:

$$\alpha = \arg \min_i \gamma_i \leq \sigma. \quad (5)$$

In addition to the VQ indices in Φ , the VQ indices with higher occurrence numbers are selected as another set $\Theta = \{c_1, c_2, \dots, c_\beta\}$, and their corresponding index occurrence numbers are $\gamma_1, \gamma_2, \dots, \gamma_\beta$, where β is set to $L - \alpha + 1$. The remaining indices are formed as the set $\Omega = \{c_{\beta+1}, c_{\beta+2}, \dots, c_{\alpha-1}\}$.

In the following, VQ indices from the two sets Φ and Θ are exploited to construct β index groups through an iterative strategy. In detail, the β index groups are emptied initially. Then, the index with the highest occurrence number, denoted as $c_g^{(j)}$, in the current set Θ and the index with the lowest occurrence number, denoted as $c_s^{(j)}$, in the

current set Φ are selected to form one index group $\{c_g^{(j)}, c_s^{(j)}\}$, $j = 1, 2, \dots, \beta$, and the two indices, $c_g^{(j)}$ and $c_s^{(j)}$, are removed from Θ and Φ , respectively. According to the above way, β index groups can be constructed iteratively until the two sets Θ and Φ become empty.

3.2.2. Data Embedding. In order to guarantee the reversibility of original VQ-encoded image on the receiver side, side information should be recorded for the indices in Φ whose occurrence numbers are not zero, i.e., $c_i \in \Phi$ and $\gamma_i \neq 0$, $i \in \{\alpha, \alpha + 1, \dots, L\}$. In detail, for the j th index group $\{c_g^{(j)}, c_s^{(j)}\}$, $j = 1, 2, \dots, \beta$, we first utilize $\log_2(MN/n^2)$ bits to sequentially represent the occurrence number of the index $c_s^{(j)}$ in \mathbf{T}_e ; if the occurrence number of the index $c_s^{(j)}$ in \mathbf{T}_e , i.e., γ_i corresponding to $c_i \in \Phi$, does not equal 0, we should further utilize $\gamma_i \cdot \log_2(MN/n^2)$ bits to record the position information of the γ_i indices in \mathbf{T}_e . Thus, the length of side information is:

$$\rho = \left(\beta + \sum_{i=\alpha}^L \gamma_i \right) \cdot \log_2 \left(\frac{MN}{n^2} \right). \quad (6)$$

We compress the side information by run-length coding and concatenate the compressed side information with the additional data \mathbf{w} to be embedded together as w' after scrambling with the data-hiding key K_h . During data embedding, for each index group $\{c_g^{(j)}, c_s^{(j)}\}$, if the occurrence number of the index $c_s^{(j)}$ in \mathbf{T}_e is not equal to 0, data hider should replace all the index values $c_s^{(j)}$ in \mathbf{T}_e with the indices c_m that can be randomly selected from the set Ω , and the modified index table is denoted as \mathbf{T}'_e . Then, each VQ index in \mathbf{T}'_e that is equal to the index $c_g^{(j)}$ with higher occurrence number in one of the β index groups, i.e., $\{c_g^{(j)}, c_s^{(j)}\}$, can be embedded with one binary bit. In detail, data hider scans the VQ indices in \mathbf{T}'_e with the raster-scanning order, and if the current scanning index $T_{x,y}$ is equal to $c_g^{(j)}$ in the j th index group ($j = 1, 2, \dots, \beta$), one binary bit w_i from w' can be embedded by:

$$T'_{x,y} = \begin{cases} c_g^{(j)}, & \text{if } w_i = 0, \\ c_s^{(j)}, & \text{if } w_i = 1, \end{cases} \quad (7)$$

where $T'_{x,y}$ denotes the marked VQ index. In other words, for the current scanning index $T_{x,y}$ belonging to the set Θ , if the to-be-embedded bit is 0, $T_{x,y}$ remains unchanged, otherwise, $T_{x,y}$ is changed to its corresponding index with lower occurrence number in the same index group.

After all VQ indices, belonging to Θ , in \mathbf{T}'_e finish the above procedure, we can obtain a marked, encrypted index table \mathbf{T}_{ew} . Then, \mathbf{T}_{ew} and \mathbf{C}_e are transmitted to the receiver for data extraction and image recovery. Note that the β index groups should also be sent to the receiver as auxiliary data \mathfrak{R} .

3.3. Data Extraction and Image Recovery. When the receiver obtains the marked, encrypted index table \mathbf{T}_{ew} , the encrypted codebook \mathbf{C}_e and the auxiliary data \mathfrak{R} , data extraction and image recovery can be conducted. There are three scenarios:

(1) if the receiver only has the data-hiding key K_h , the additional data \mathbf{w} can be extracted correctly; (2) if the receiver only has the encryption key K_e , a directly decrypted index table \mathbf{T}_d , which is similar to the original index table \mathbf{T} can be obtained; and (3) if the receiver has both K_e and K_h , additional data \mathbf{w} and original index table \mathbf{T} can both be recovered with no error. Details are presented as follows.

3.3.1. Data Extraction. First, the two index sets Θ and Φ corresponding to higher and lower occurrence numbers can be easily obtained based on the auxiliary data \mathfrak{R} representing the β index groups. Then, during scanning the index table \mathbf{T}_{ew} in the raster-scanning order, according to the current scanning index $T'_{x,y}$ belonging to Θ or Φ , the embedded bit w'_i can be extracted sequentially, see equation (8).

$$w'_i = \begin{cases} 0, & \text{if } T'_{x,y} \in \Theta, \\ 1, & \text{if } T'_{x,y} \in \Phi, \\ \text{no data extracted,} & \text{otherwise.} \end{cases} \quad (8)$$

Concatenating all extracted bits w'_i , the embedded data w' can be obtained correctly. Then, the receiver inversely scrambles w' through the data-hiding key K_h , and parses the ρ -bits side information from w' , thus, the remaining part is the extracted additional data \mathbf{w} .

3.3.2. Image Decryption. If the receiver only has the encryption key $K_e = \{K_e^{(1)}, K_e^{(2)}\}$, he/she can first generate the sequence of pseudo-random bits $S_{i,j,k}$ ($i = 1, 2, \dots, L$, $j = 1, 2, \dots, n^2$, $k = 1, 2, \dots, 8$) by $K_e^{(1)}$, which is the same with the one on the content-owner side. Through decrypting based on XOR operation, the decrypted codebook \mathbf{C}_d can be obtained, which is exactly the same as the original codebook \mathbf{C} .

On the other hand, the receiver scans the marked, encrypted index table \mathbf{T}_{ew} , and if the current scanning index $T'_{x,y}$ is equal to $c_s^{(j)}$ in one of the β index groups ($j = 1, 2, \dots, \beta$), $T'_{x,y}$ is modified as the corresponding $c_g^{(j)}$ in the same group. After all indices in \mathbf{T}_{ew} are performed, a new index table \mathbf{T}'_e can be produced. Then, through decrypting \mathbf{T}'_e based on permutation with $K_e^{(2)}$, a directly decrypted index table \mathbf{T}_d , which is similar to the original index table \mathbf{T} , can be obtained. If required, a directly decrypted image \mathbf{I}_d can also be acquired through decoding the index table \mathbf{T}_d by the VQ codebook \mathbf{C} .

As we know, the side information records the numbers and the positions for the indices in Φ with nonzero occurrence numbers, and these indices are replaced by c_m randomly selected in Ω during data embedding. Therefore, due to the unavailability of the data-hiding key K_h , the side information cannot be parsed from w' , which means these recorded indices cannot be recovered from c_m in Ω to $c_s^{(j)}$ in Φ . In other words, without K_h , \mathbf{T}'_e cannot be recovered to \mathbf{T}_e perfectly, and \mathbf{T}_d is not exactly the same with \mathbf{T} .

3.3.3. Image Recovery. If both encryption key K_e and data-hiding key K_h are available, the receiver can obtain the index table \mathbf{T}'_e and can also parse the ρ -bits side information and

the additional data \mathbf{w} from the extracted data w' with K_h . According to the parsed side information, the receiver can know the detailed numbers and positions for the indices in Φ with nonzero occurrence numbers, which are replaced by c_m randomly selected in Ω during data embedding. Thus, through scanning the index table \mathbf{T}'_e , the indices at the positions indicated in the side information can be restored from c_m to the corresponding index $c_s^{(j)}$ (whose occurrence number is not zero) in the index group $\{c_g^{(j)}, c_s^{(j)}\}$. As a result, the index table \mathbf{T}'_e is recovered to \mathbf{T}_e perfectly, and after decrypting \mathbf{T}_e based on permutation with $K_e^{(2)}$, the original index table \mathbf{T} can be recovered reversibly. As described previously, the original codebook \mathbf{C} can be obtained through decrypting \mathbf{C}_d with $K_e^{(1)}$ based on XOR operation. Therefore, the additional data \mathbf{w} , original index table \mathbf{T} , and original codebook \mathbf{C} can all be acquired when both K_e and K_h are available. If required, the original VQ-decoded image \mathbf{I} can also be acquired through decoding \mathbf{T} by \mathbf{C} .

4. Performance Optimization

The proposed scheme described in Section 3 can be considered as the baseline, which can be further optimized on the performance of hiding capacity during additional-data embedding. The optimization mainly focuses on adaptively adjusting the number of indices with lower occurrence numbers in index groups. Note that encryption operation on the content-owner side is unchanged. Details are given as follows.

4.1. Generalized Index Grouping. In the optimized scheme, during index grouping on the data-hider side, we define that, in each group, the number of indices with higher occurrence numbers is fixed as 1, while the number of indices with lower occurrence numbers should be $2^\nu - 1$, where ν is a variable integer satisfying:

$$\nu \in \{1, 2, \dots, \lfloor \log_2(L - \alpha + 1) \rfloor\}. \quad (9)$$

Thus, one index $c_g^{(j)}$ with higher occurrence numbers selected from $\Theta = \{c_1, c_2, \dots, c_\beta\}$ and $2^\nu - 1$ indices, $c_s^{(j,1)}, c_s^{(j,2)}, \dots, c_s^{(j,2^\nu-1)}$, with lower occurrence numbers selected from $\Phi = \{c_{\alpha_1}, c_{\alpha_1+1}, \dots, c_L\}$ can be constructed as one index group, i.e., $\{c_g^{(j)}, c_s^{(j,1)}, c_s^{(j,2)}, \dots, c_s^{(j,2^\nu-1)}\}$. Note that if ν is a constant equaling 1 for all groups, the optimized scheme is just the baseline proposed in Section 3. In addition, different from the baseline scheme in Section 3, in the optimized scheme, the number β of the indices with higher occurrence numbers in $\Theta = \{c_1, c_2, \dots, c_\beta\}$ may not be equal to the number $(L - \alpha + 1)$ of the indices with lower occurrence numbers in $\Phi = \{c_{\alpha_1}, c_{\alpha_1+1}, \dots, c_L\}$.

We consider the index groups including the same number $2^\nu - 1$ of lower occurrence indices as the same (i.e., the ν th) type of index groups, and a coefficient μ_ν is defined to represent the number of index groups belonging to the ν th type, $\nu = 1, 2, \dots, \log_2(L - \alpha + 1)$. A coefficient vector $\boldsymbol{\mu}$ can be given for different types of index groups, see equation (10). Table 1 lists the detailed information for different types of index groups.

TABLE 1: Details of different types of existing index groups.

Group type	$\nu = \log_2(L - \alpha + 1)$...	$\nu = 2$	$\nu = 1$
The number of c_g in group	1	...	1	1
The number of c_s in group	$2^\nu - 1$...	$2^2 - 1$	$2^1 - 1$
Group coefficient	μ_ν	...	μ_2	μ_1
Embedding ability (bits)	$\log_2(L - \alpha + 1)$...	2	1

$$\boldsymbol{\mu} = [\mu_1, \mu_2, \dots, \mu_{\lfloor \log_2(L - \alpha + 1) \rfloor}]. \quad (10)$$

The generalized index grouping should satisfy the following two relationships:

$$L - \alpha + 1 = \sum_{\nu=1}^{\lfloor \log_2(L - \alpha + 1) \rfloor} \mu_\nu \cdot (2^\nu - 1), \quad (11)$$

$$\beta = \sum_{\nu=1}^{\lfloor \log_2(L - \alpha + 1) \rfloor} \mu_\nu. \quad (12)$$

Equation (11) implies that $\mu_\nu \cdot (2^\nu - 1)$ represents the number of indices from Φ belonging to the ν th type of index group, and in all $\log_2(L - \alpha + 1)$ types of index groups, the total number of indices from Φ should be equal to $L - \alpha + 1$. On the other hand, Equation (12) guarantees that each index group has one index from Θ .

For intuitive description, we present an example of generalized index grouping in Figure 4. Figure 4(a) shows the sorted indices and their corresponding code words ($L = 128$), which are sorted in the descending order according to occurrence numbers of indices within index table \mathbf{T}_e . Here, we set the parameter σ in equation (4) to 1, thereby, α can be derived as 116. Thus, we can obtain $128 - 116 + 1 = 13$ indices with lower occurrence numbers in Φ , i.e., $\{c_{116}, c_{117}, \dots, c_{128}\}$. Then, as shown in Figure 4(b), after generalized index grouping, there are three types of index groups (including five index groups totally), corresponding to $\nu = 3$ ($\mu_3 = 1$), $\nu = 2$ ($\mu_2 = 1$), $\nu = 1$ ($\mu_1 = 3$), respectively. The value of β can also be obtained as $1 + 1 + 3 = 5$ through equation (12), which means that five indices with higher occurrence numbers are included in Θ , i.e., $\{c_1, c_2, \dots, c_5\}$. In detail, $\{c_1, c_{122}, c_{123}, c_{124}, c_{125}, c_{126}, c_{127}, c_{128}\}$ belongs to the third type of index group ($\nu = 3, \mu_3 = 1$); $\{c_2, c_{119}, c_{120}, c_{121}\}$ belongs to the second type of index group ($\nu = 2, \mu_2 = 1$); $\{c_3, c_{118}\}$, $\{c_4, c_{117}\}$, and $\{c_5, c_{116}\}$ belong to the first type of index groups ($\nu = 1, \mu_1 = 3$). The coefficient vector $\boldsymbol{\mu}$ is equal to $[1, 1, 3]$. Table 2 summarizes the index grouping information for the example in Figure 4.

4.2. Multiple-Bits Embedding. Similar with the baseline scheme described in Section 3, we also need to sequentially record the occurrence numbers of the indices belonging to Φ and their positions (if existing) in \mathbf{T}_e as the ρ -bits side information, see equation (6). Then, side information can be compressed through run-length coding and be concatenated with the additional data \mathbf{w} as w' after scrambling with the data-hiding key K_h . During data embedding, for any index in Φ whose occurrence number is not equal to 0, data hider should replace this index value in \mathbf{T}_e with the index c_m that

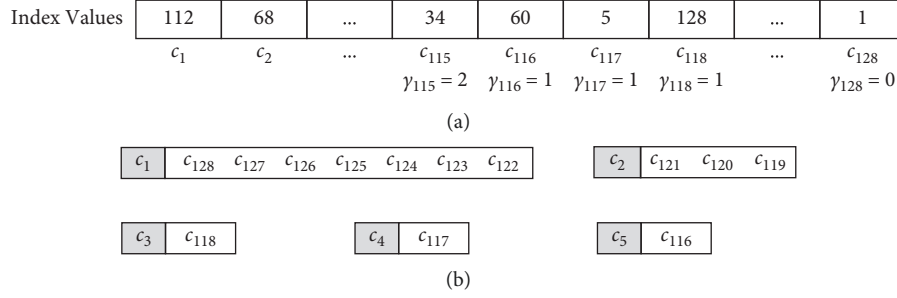


FIGURE 4: An example of generalized index grouping. (a) The sorted indices, (b) A result of index grouping.

TABLE 2: Index grouping information for the example in Figure 4.

Group Type	$\nu = 3$	$\nu = 2$	$\nu = 1$
The number of c_g in group	1	1	1
The number of c_s in group	7	3	1
Group coefficient	1	1	3
Embedding ability (bits)	3	2	1

can be randomly selected from the set Ω , and the modified index table is also denoted as \mathbf{T}'_e . Then, each VQ index in \mathbf{T}'_e that is equal to the index $c_g^{(j)}$ with higher occurrence number in the generalized index group, i.e., $\{c_g^{(j)}, c_s^{(j,1)}, c_s^{(j,2)}, \dots, c_s^{(j,2^{\nu}-1)}\}$, can be embedded with ν binary bits ($j = 1, 2, \dots, \beta$). In detail, data hider scans the VQ indices in \mathbf{T}'_e with the raster-scanning order, and if the current scanning index $T_{x,y}$ is equal to $c_g^{(j)}$ in the j th index group, multiple bits, $\{w_i, w_{i+1}, \dots, w_{i+\nu-1}\}$, from \mathbf{w}' can be embedded by:

$$T'_{x,y} = \begin{cases} c_g^{(j)}, & \text{if } \tau_i = 0, \\ c_s^{(j,1)}, & \text{if } \tau_i = 1, \\ \dots & \dots \\ c_s^{(j,2^{\nu}-1)}, & \text{if } \tau_i = 2^{\nu} - 1, \end{cases} \quad (13)$$

where τ_i denotes the decimal value of the current ν bits $\{w_i, w_{i+1}, \dots, w_{i+\nu-1}\}$ for embedding, and $T'_{x,y}$ denotes the marked VQ index embedded with the ν bits. Equation (13) means that, for the current scanning index $T_{x,y}$ belonging to the set Θ , if the decimal value τ_i of the current to-be-embedded ν bits is 0, $T_{x,y}$ remains unchanged, otherwise, $T_{x,y}$ is changed to the τ_i th corresponding index with a lower occurrence number in the same index group. After all VQ indices, belonging to Θ , in \mathbf{T}'_e are scanned and performed with above operations orderly, the procedure of multiple-bits embedding is finished and the marked, encrypted index table \mathbf{T}_{ew} can be acquired.

Continuing the example in Figure 4, since $\{c_1, c_{122}, c_{123}, c_{124}, c_{125}, c_{126}, c_{127}, c_{128}\}$ belongs to the third type of index group ($\nu = 3$), three binary bits can be embedded when the current scanning index is c_1 ; since $\{c_2, c_{119}, c_{120}, c_{121}\}$ belongs to the second type of index group ($\nu = 2$), two binary bits can be embedded when the current scanning index is c_2 ; since $\{c_3, c_{118}\}$, $\{c_4, c_{117}\}$ and $\{c_5, c_{116}\}$ belong to the first type of index group ($\nu = 1$), one binary bit can be embedded when the current scanning index is c_3 , c_4 , or c_5 . It can be inferred

that, after data embedding, the occurrence numbers of c_1, c_2, c_3, c_4 and c_5 in the index table are decreased because a portion of them are changed to the indices with lower occurrence number in their corresponding index groups.

Obviously, if index groups are determined, the hiding capacity ζ of the optimized scheme can be calculated. The occurrence numbers for the VQ indices with higher occurrence numbers, $\{c_1, c_2, \dots, c_\beta\}$, in Θ are denoted as $\mathbf{f} = [\gamma_1, \gamma_2, \dots, \gamma_\beta]$. According to the coefficient vector $\boldsymbol{\mu}$ in equation (10), the occurrence number vector \mathbf{f} can be transformed to a $1 \times \log_2(L - \alpha + 1)$ vector $\boldsymbol{\eta} = [\eta_1, \eta_2, \dots, \eta_{\log_2(L - \alpha + 1)}]$:

$$\eta_\nu = \begin{cases} 0, & \mu_\nu = 0, \\ \sum_{i=\kappa_\nu+1}^{\kappa_\nu+\mu_\nu} \gamma_i, & \mu_\nu \neq 0, \end{cases} \quad \nu = 1, 2, \dots, \lfloor \log_2(L - \alpha + 1) \rfloor, \quad (14)$$

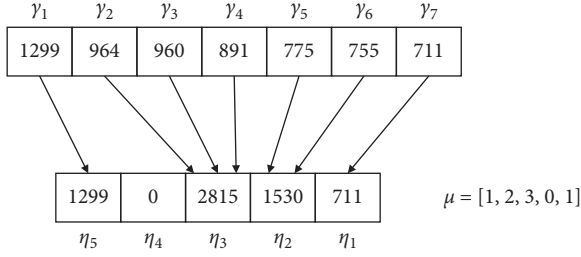
$$\kappa_\nu = \begin{cases} 0, & \nu = \lfloor \log_2(L - \alpha + 1) \rfloor, \\ \sum_{j=\nu+1}^{\lfloor \log_2(L - \alpha + 1) \rfloor} \mu_j, & \nu < \lfloor \log_2(L - \alpha + 1) \rfloor, \end{cases} \quad (15)$$

where η_ν denotes the number of indices with higher occurrence numbers belonging to the ν th type of index group, $\nu = 1, 2, \dots, \log_2(L - \alpha + 1)$. Note that, the lengths of the two vectors $\boldsymbol{\mu}$ and $\boldsymbol{\eta}$ are equal. An example of transform procedure from \mathbf{f} to $\boldsymbol{\eta}$ is given in Figure 5. Assume that the occurrence number vector is $\mathbf{f} = [1299, 964, 960, 891, 775, 755, 711]$ and the coefficient vector is $\boldsymbol{\mu} = [1, 2, 3, 0, 1]$, respectively. With the assistance of $\boldsymbol{\mu}$, the vector $\boldsymbol{\eta}$ can be obtained as $[711, 775 + 755, 964 + 960 + 891, 0, 1299] = [711, 1530, 2815, 0, 1299]$.

Therefore, based on the above descriptions, the hiding capacity ζ of the proposed scheme can be obtained as:

$$\zeta = \sum_{\nu=1}^{\lfloor \log_2(L - \alpha + 1) \rfloor} \nu \cdot \eta_\nu, \quad (16)$$

The operations on the receiver side, including data extraction, image decryption, and recovery after receiving the marked, encrypted index table \mathbf{T}_{ew} , the encrypted codebook \mathbf{C}_e , and the auxiliary data \mathfrak{R} of the β index group, have minor differences with those described in Section 3. As for data extraction, \mathbf{T}_{ew} is first scanned in the raster-scanning order, and if the current scanning index $T'_{x,y}$ is equal to one of the indices in an index group, i.e., $\{c_g^{(j)}, c_s^{(j,1)}, c_s^{(j,2)}, \dots, c_s^{(j,2^{\nu}-1)}\}$, $j = 1, 2, \dots, \beta$, ν binary bits can be

FIGURE 5: An example of transform procedure from \mathbf{f} to $\boldsymbol{\eta}$.

extracted according to equation (13), and then, the side information and additional data can be parsed with K_h . As for image decryption, $T_{x,y}'$ is modified as corresponding to the high-occurrence index $c_g^{(j)}$ in the same group, thus, \mathbf{T}_{ew} is changed as \mathbf{T}'_e after all indices in \mathbf{T}_{ew} are scanned. Then, the original VQ codebook \mathbf{C} and the directly decrypted index table \mathbf{T}_d are produced through decrypting \mathbf{C}_e and \mathbf{T}'_e with $K_e^{(1)}$ and $K_e^{(2)}$, respectively. The directly decrypted image \mathbf{I}_d can be obtained by decoding \mathbf{T}_d by \mathbf{C} . As for image recovery, the index table \mathbf{T}'_e should be first recovered to \mathbf{T}_e with the assistance of side information. As we know, side information sequentially records the occurrence numbers and the positions (if existing) for the $L - \alpha + 1$ indices belonging to Φ with lower occurrence numbers in \mathbf{T}_e , among which those indices with nonzero occurrence numbers are replaced by c_m randomly selected in Ω during data embedding. Therefore, with the parsed side information, the receiver can sequentially restore c_m back to the corresponding index with nonzero, lower occurrence numbers in $\{c_\alpha, c_{\alpha+1}, \dots, c_L\}$. As a result, \mathbf{T}'_e can be recovered to \mathbf{T}_e perfectly, and after decrypting \mathbf{T}_e , original index table \mathbf{T} can be obtained. Finally, original VQ-decoded image \mathbf{I} can also be acquired through decoding \mathbf{T} by \mathbf{C} .

4.3. Hiding Capacity Optimization. It should be noticed that there possibly exist multiple coefficient vectors $\boldsymbol{\mu}$ that can satisfy the two relationships in equations (11) and (12), and different coefficient vectors $\boldsymbol{\mu}$ may lead to different hiding capacities. Therefore, by finding the optimal coefficient vector, hiding capacity of the proposed scheme can be further optimized.

Suppose that there are λ different coefficient vectors, $\boldsymbol{\mu}^{(1)}$, $\boldsymbol{\mu}^{(2)}$, \dots , $\boldsymbol{\mu}^{(\lambda)}$, satisfying equations (11) and (12), which can be represented by a matrix \mathbf{U} sized $\lambda \times \log_2(L - \alpha + 1)$:

$$\mathbf{U} = \begin{bmatrix} \boldsymbol{\mu}^{(1)} \\ \boldsymbol{\mu}^{(2)} \\ \vdots \\ \boldsymbol{\mu}^{(\lambda)} \end{bmatrix} = \begin{bmatrix} \mu_1^{(1)} & \mu_2^{(1)} & \cdots & \mu_{\lfloor \log_2(L-\alpha+1) \rfloor}^{(1)} \\ \mu_1^{(2)} & \mu_2^{(2)} & \cdots & \mu_{\lfloor \log_2(L-\alpha+1) \rfloor}^{(2)} \\ \vdots & \vdots & \cdots & \vdots \\ \mu_1^{(\lambda)} & \mu_2^{(\lambda)} & \cdots & \mu_{\lfloor \log_2(L-\alpha+1) \rfloor}^{(\lambda)} \end{bmatrix}, \quad (17)$$

where $\boldsymbol{\mu}^{(i)} = [\mu_1^{(i)}, \mu_2^{(i)}, \dots, \mu_{\lfloor \log_2(L-\alpha+1) \rfloor}^{(i)}]$, $i = 1, 2, \dots, \lambda$. According to equations (14) and (15), we can know that each row in the matrix \mathbf{U} , i.e., $\boldsymbol{\mu}^{(i)}$, corresponds to a vector $\boldsymbol{\eta}^{(i)}$ based on the occurrence number vector $\mathbf{f} = [\gamma_1, \gamma_2, \dots, \gamma_\beta]$. Thus, the λ vectors, $\boldsymbol{\eta}^{(1)}$, $\boldsymbol{\eta}^{(2)}$, \dots , $\boldsymbol{\eta}^{(\lambda)}$, can form a matrix $\boldsymbol{\Gamma}$ that has the same size with \mathbf{U} :

$$\boldsymbol{\Gamma} = \begin{bmatrix} \boldsymbol{\eta}^{(1)} \\ \boldsymbol{\eta}^{(2)} \\ \vdots \\ \boldsymbol{\eta}^{(\lambda)} \end{bmatrix} = \begin{bmatrix} \eta_1^{(1)} & \eta_2^{(1)} & \cdots & \eta_{\lfloor \log_2(L-\alpha+1) \rfloor}^{(1)} \\ \eta_1^{(2)} & \eta_2^{(2)} & \cdots & \eta_{\lfloor \log_2(L-\alpha+1) \rfloor}^{(2)} \\ \vdots & \vdots & \cdots & \vdots \\ \eta_1^{(\lambda)} & \eta_2^{(\lambda)} & \cdots & \eta_{\lfloor \log_2(L-\alpha+1) \rfloor}^{(\lambda)} \end{bmatrix}, \quad (18)$$

where $\boldsymbol{\eta}^{(i)} = [\eta_1^{(i)}, \eta_2^{(i)}, \dots, \eta_{\lfloor \log_2(L-\alpha+1) \rfloor}^{(i)}]$, $i = 1, 2, \dots, \lambda$. Then, based on equation (16), we can obtain λ values, $\zeta^{(1)}$, $\zeta^{(2)}$, \dots , $\zeta^{(\lambda)}$, of the hiding capacity:

$$\zeta^{(i)} = \mathbf{r} \cdot (\boldsymbol{\eta}^{(i)})^T, \quad (19)$$

where \mathbf{r} denotes the row vector $[1, 2, \dots, \log_2(L - \alpha + 1)]$, and $\zeta^{(i)}$ is the hiding capacity corresponding to the coefficient vector $\boldsymbol{\mu}^{(i)}$ for index groups, $i = 1, 2, \dots, \lambda$. With equation (20), the optimal coefficient vector can be found as $\boldsymbol{\mu}^{(i^*)}$ by dynamic programming, which means that the optimal result of generalized index grouping is determined. Finally, the largest hiding capacity of the proposed scheme after optimization can be acquired as $\zeta^{(i^*)}$.

$$i^* = \arg \max_i \zeta^{(i)}, \quad (20)$$

subject to $i \in \{1, 2, \dots, \lambda\}$.

5. Experimental Results and Analysis

In order to demonstrate the effectiveness and superiority of our scheme, experiments were conducted on a large number of VQ-encoded images, and the environment of our experiments was based on a personal computer with a 3.20 GHz Intel i5 processor, 4.00 GB memory, Windows 10 operating system, and Matlab R2016a. In the following, results of the proposed scheme, including the reversibility, hiding capacity ζ , and visual quality of directly decrypted image \mathbf{I}_d , are first given. Then, the influences of parameter σ on the performances are analyzed. Finally, comparisons with state-of-the-art schemes are discussed.

5.1. Results of the Proposed Scheme

5.1.1. Reversibility. Figure 6(a) shows an original VQ-decoded image \mathbf{I} for *Lena* sized 512×512 , the length L of corresponding VQ codebook \mathbf{C} is 256. In this experiment, the parameter σ in equation (6) was set as 1, and α was equal to 226 accordingly. The VQ-decoded, encrypted image with index permutation and codebook encryption is shown in Figure 6(b), which is the result through decoding \mathbf{T}_e with \mathbf{C}_e . It can be observed that the contents of the original VQ-decoded image \mathbf{I} are effectively masked after encryption. Figure 6(c) shows the VQ decoded, encrypted image after data embedding, which is the result through decoding \mathbf{T}_{ew} with \mathbf{C}_e . The hiding capacity ζ was 12954 bits. Figure 6(d) is the directly decrypted image \mathbf{I}_d for Figure 6(c), which is the result through decoding \mathbf{T}_d with \mathbf{C} . PSNR of the directly decrypted result \mathbf{I}_d in Figure 6(d) is 41.80 dB with respect to the original VQ-decoded image \mathbf{I} in Figure 6(a). Recovered image, i.e., the result through decoding \mathbf{T} with \mathbf{C} , is exactly

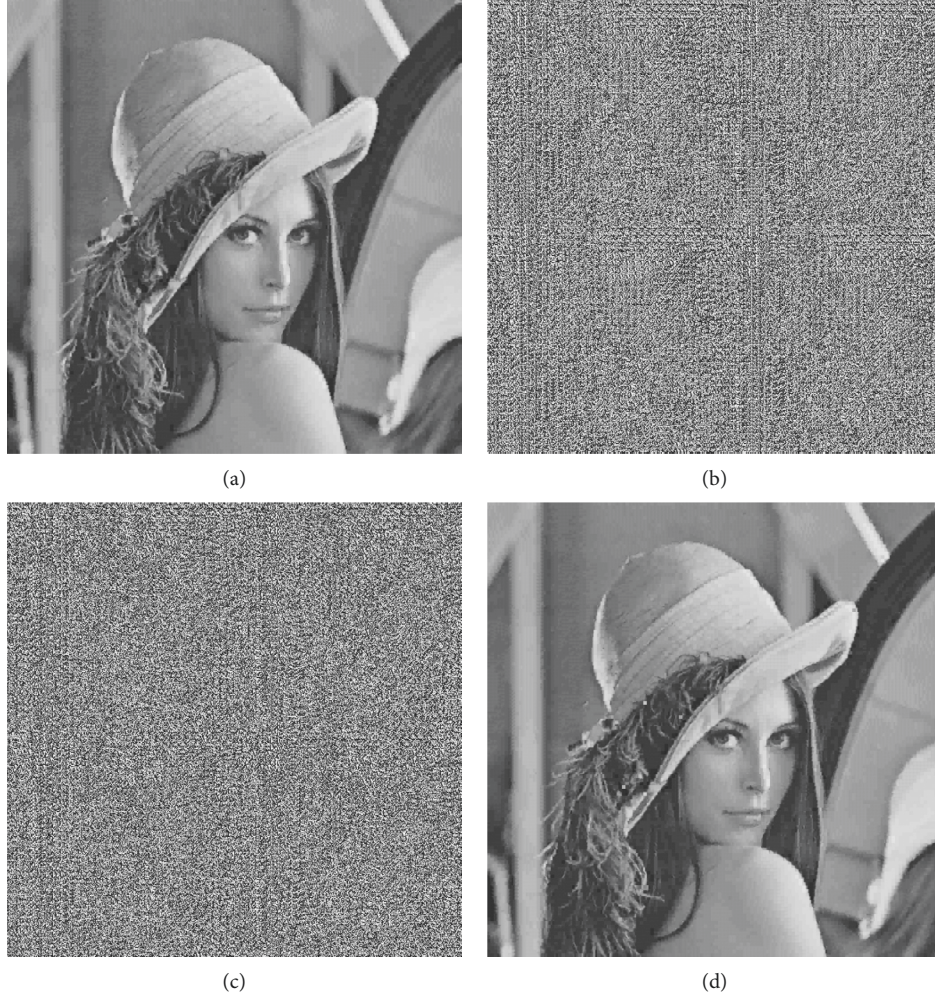


FIGURE 6: Results of the proposed scheme for image *Lena*. (a) Original VQ-image with decoding (T) by C. (b) Encrypted VQ-image with decoding T_e by C_e . (c) Marked, encrypted VQ-image with decoding T_{ew} by C_e ($\zeta = 12954$ bits). (d) Directly decrypted VQ-image with decoding T_d by (C) (PSNR = 41.80 dB).

the same as I , i.e., PSNR = $+\infty$, which demonstrates the reversibility of our scheme.

5.1.2. Hiding Capacity. Figure 7 shows four standard test images sized 512×512 , including *Airplane*, *Baboon*, *Lena*, and *Peppers*. VQ compression was conducted for these four images, and the sizes L of the adopted codebooks can be 128, 256, 512, and 1024. Figure 8 shows hiding capacities ζ of our scheme for the four VQ-encoded images after encryption ($\sigma = 1$), in which (a)–(d) corresponds to the VQ codebook sizes $L = 128, 256, 512,$ and 1024 , respectively. Note that the abscissa denotes the different coefficient vectors μ and the ordinate denotes corresponding hiding capacities ζ (bits). It can be observed that the codebook with larger size L can generally obtain greater hiding capacity ζ than the codebook with smaller size, since a large-size codebook can lead to more index groups for data embedding based on more VQ indices with lower occurrence numbers.

Besides the four images in Figure 7, Table 3 lists the largest hiding capacities with the optimal coefficient vectors

$\mu^{(i^*)}$ of our scheme for more images. Actually, for different images, the hiding capacity of our scheme is mainly related with two aspects: (1) the value of $(L - \alpha + 1)$ under a given parameter σ , i.e., the number of the low-occurrence indices $\{c_\alpha, c_{\alpha+1}, \dots, c_L\}$ and (2) the occurrence number vector $\mathbf{f} = [\gamma_1, \gamma_2, \dots, \gamma_\beta]$, i.e., occurrence numbers of high-occurrence indices $\{c_1, c_2, \dots, c_\beta\}$ in T_e , see equations (14)–(16). Figure 9 shows the histograms of VQ indices that are sorted in the descending order according to occurrence numbers for the two images *Airplane* and *Baboon* ($L = 512$). We can clearly observe that the index histogram distribution of *Airplane* is more concentrated than that of *Baboon*, which means that the number of low-occurrence indices $\{c_\alpha, c_{\alpha+1}, \dots, c_L\}$ and the occurrence numbers of high-occurrence indices $\{c_1, c_2, \dots, c_\beta\}$ for *Airplane* are greater than those of *Baboon*. Correspondingly, it can be found from Table 3 that the hiding capacity for *Airplane* is significantly greater than that of *Baboon*. In addition, we also conducted experiments on the UCID image database [45], which consists of 1338 distinct images with the sizes of 512×384 and 384×512 , see the last row of Table 3. For color images in the UCID

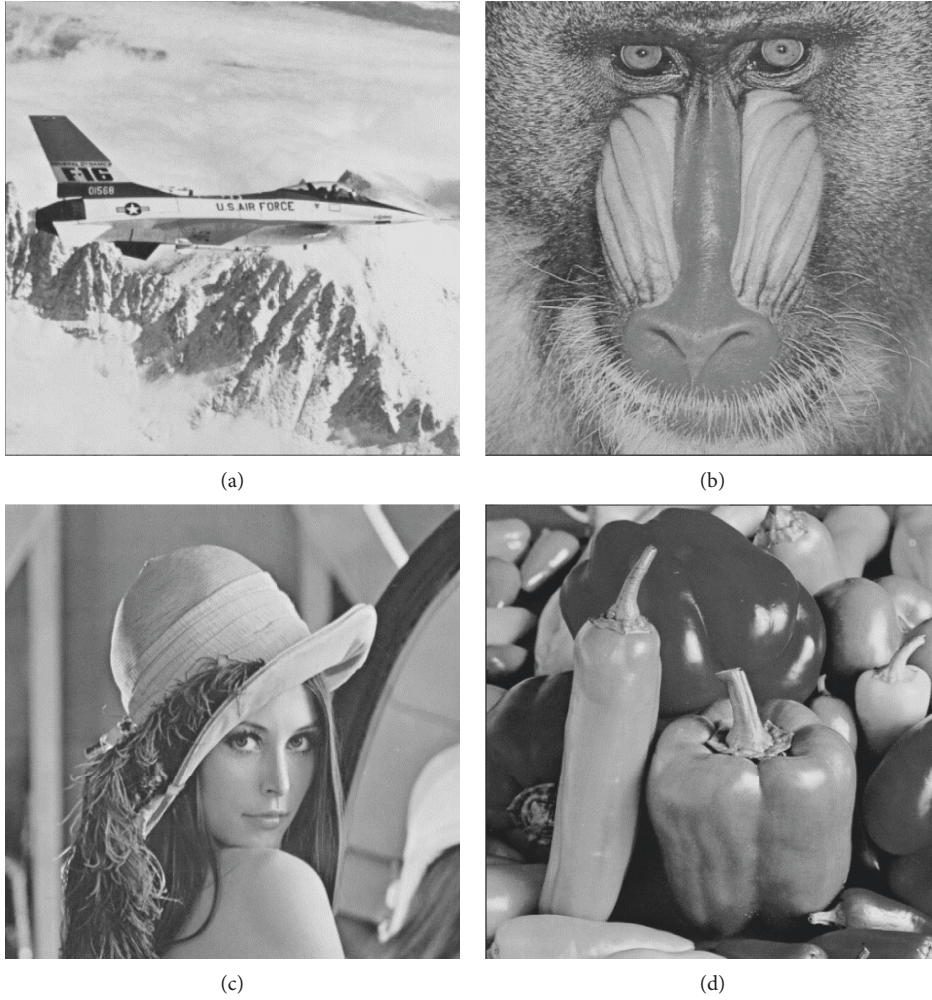


FIGURE 7: Four standard test images. (a) Airplane. (b) Baboon. (c) Lena. (d) Peppers.

database, the luminance components were applied for test, and the average hiding capacities are 6328 bits, 14649 bits, 21819 bits, and 29643 bits, when L is equal to 128, 256, 512, and 1024, respectively.

5.1.3. Visual Quality of Directly-Decrypted Image. As described previously, on the receiver side, the marked, encrypted index table \mathbf{T}_{ew} , is scanned in raster-scanning order, and if the current scanning index $T_{x,y}$ is equal to one of low-occurrence indices in an index group, i.e., $\{c_s^{(j,1)}, c_s^{(j,2)}, \dots, c_s^{(j,2^j-1)}\}$, is equal to one of low-occurrence is modified as the corresponding the high-occurrence index $c_g^{(j)}$ in the same group, $j = 1, 2, \dots, \beta$. After all indices in \mathbf{T}_{ew} are performed, \mathbf{T}_{ew} is changed as \mathbf{T}'_e . Then, after decrypting \mathbf{T}'_e , a directly-decrypted index table \mathbf{T}_d is generated, and the directly-decrypted image \mathbf{I}_d is obtained with decoding \mathbf{T}_d by VQ codebook \mathbf{C} .

Visual quality of directly-decrypted image \mathbf{I}_d for the proposed scheme is given in Table 4. As it reveals, PSNR values of directly-decrypted images decrease when the size L of VQ codebook increases. When the codebook size L increases, there may appear more VQ indices whose

occurrence numbers are low ($\leq \sigma$) but non-zeros in the index table \mathbf{T}_e . These indices should be replaced with the index c_m randomly selected from the set Ω before data embedding, however, these indices cannot be recovered during image decryption, which causes the distortions in the directly-decrypted image \mathbf{I}_d with respect to the original VQ-decoded image \mathbf{I} . That is to say, larger codebook size L leads to more indices with non-zero, lower occurrence numbers that cannot be recovered after image decryption, thereby, lower PSNR of \mathbf{I}_d .

5.2. Performance Influence of Parameter σ . The parameter σ in equation (6) determines how many indices with lower occurrence numbers can be utilized in index grouping and data embedding; hence, the parameter σ affects the performance of hiding capacity and PSNR for the directly decrypted image, which are demonstrated in Figures 10 and 11, respectively. We can find that, with the increase of σ , the hiding capacity of our scheme increases, while the PSNR of directly decrypted image decreases. Because larger σ involves more indices with lower occurrence numbers for index grouping and data embedding, the

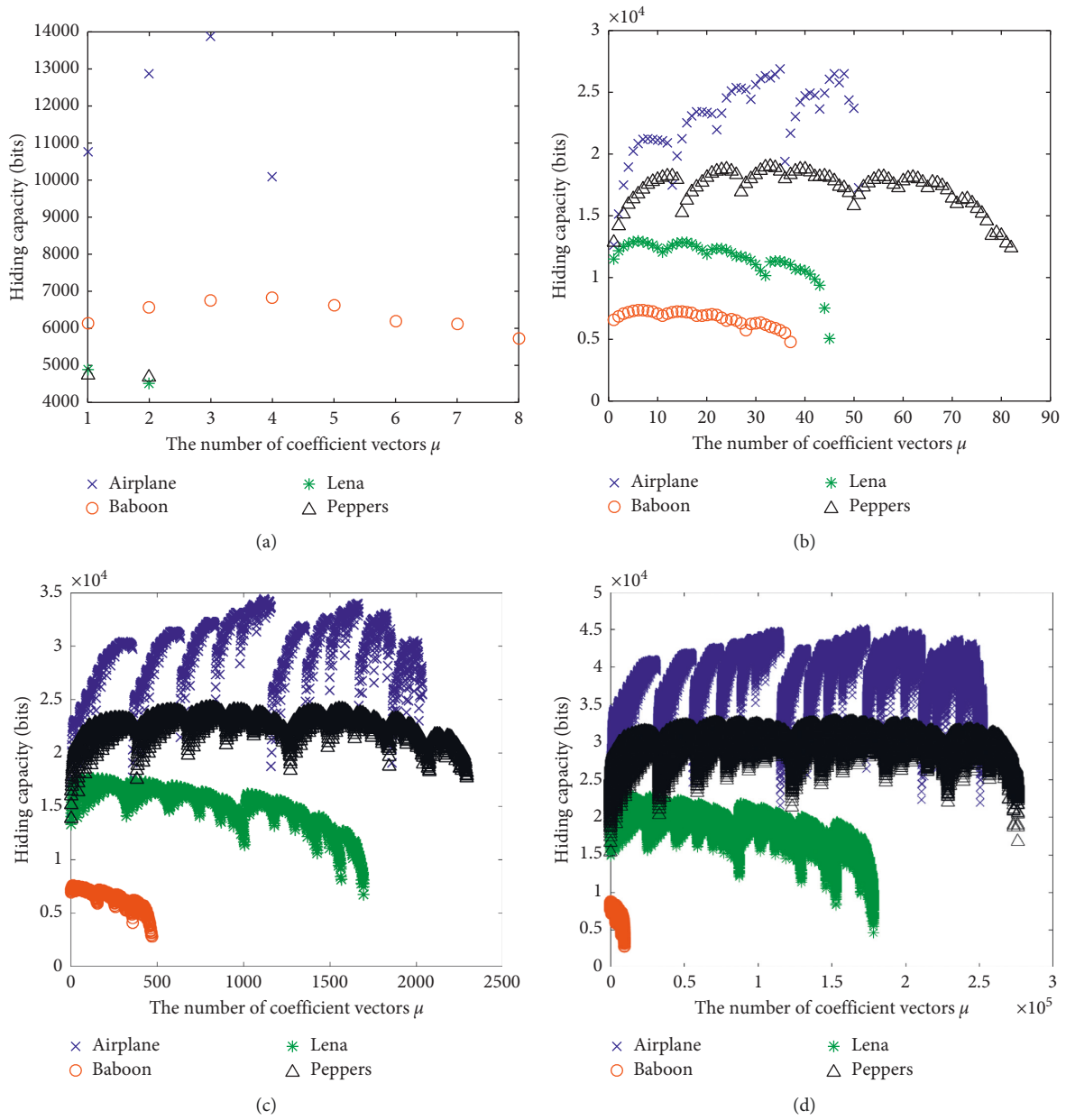


FIGURE 8: Hiding capacities with respect to different coefficient vectors μ under four kinds of codebook sizes L ($\sigma=1$). (a) $L=128$. (b) $L=256$. (c) $L=512$. (d) $L=1024$.

TABLE 3: The largest hiding capacities with the optimal coefficient vectors and $\sigma=1$ (bits).

Images	$L=128$	$L=256$	$L=512$	$L=1024$
Aerial	4902	18004	24984	32209
Baboon	6835	7360	7571	8800
Barbara	3324	10439	12566	16765
Crowd	7614	21558	24359	31859
Peppers	4738	18990	24287	32834
Airplane	13855	26876	34424	45128
Lena	4889	12954	17624	23171
UCID	6328	14649	21819	29643

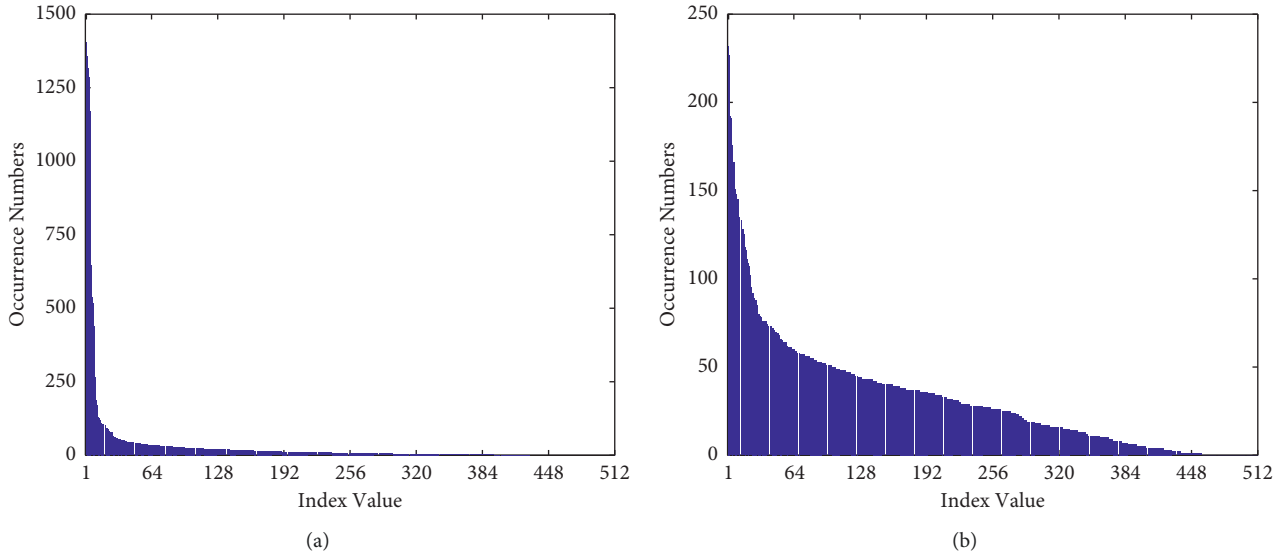
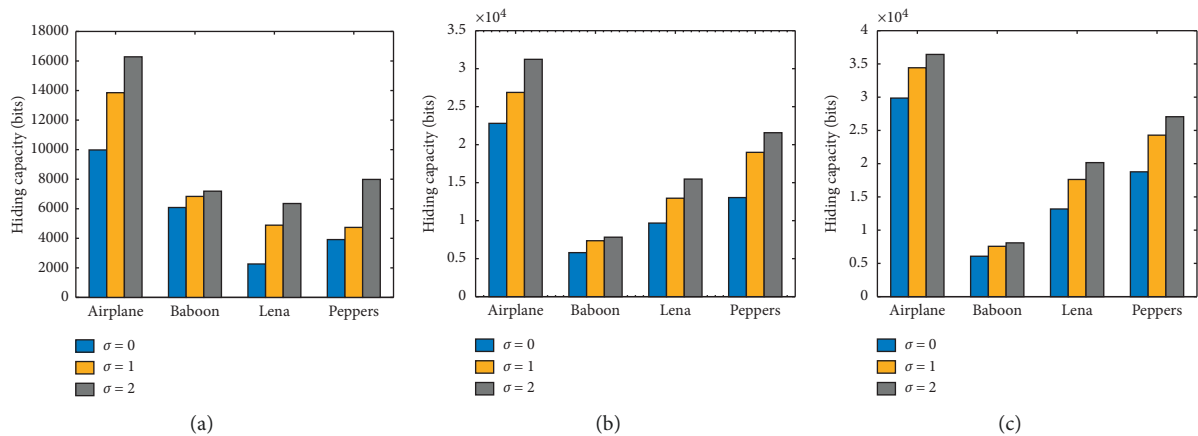

 FIGURE 9: Histograms of sorted VQ indices ($L=512$). (a) Airplane, (b) Baboon.

 TABLE 4: PSNR of directly-decrypted image I_d (dB).

Images	$L = 128$	$L = 256$	$L = 512$	$L = 1024$
Aerial	45.20	38.10	32.00	28.10
Baboon	50.40	42.80	39.50	37.00
Barbara	45.90	45.00	38.50	35.40
Crowd	62.60	43.10	38.40	33.00
Peppers	53.60	42.70	38.20	33.20
Airplane	43.90	40.30	36.80	30.50
Lena	49.10	41.80	38.60	31.60
UCID	44.91	40.73	35.36	30.13


 FIGURE 10: Hiding capacity under different values of parameter σ (bits). (a) $L = 128$. (b) $L = 256$. (c) $L = 512$.

hiding capability becomes greater with the increase of parameter σ , see Figure 10. On the other hand, when σ is equal to 0, the occurrence numbers of the $L - \alpha + 1$ indices $\{c_\alpha, c_{\alpha+1}, \dots, c_L\}$ are all zeros and no indices are required to be changed as c_m before data embedding; thus, after image decryption, original index table \mathbf{T} can be obtained since \mathbf{T}'_e is equal to \mathbf{T}_e , and the directly decrypted image I_d

is exactly the same as \mathbf{I} , i.e., PSNR = inf. However, with the increase of σ , the value of α decreases, and more indices with nonzero occurrence numbers may be included in Φ , which are required to be changed as c_m before data embedding and cannot be recovered after image decryption, thereby, leading to lower PSNR of directly decrypted image I_d , see Figure 11.

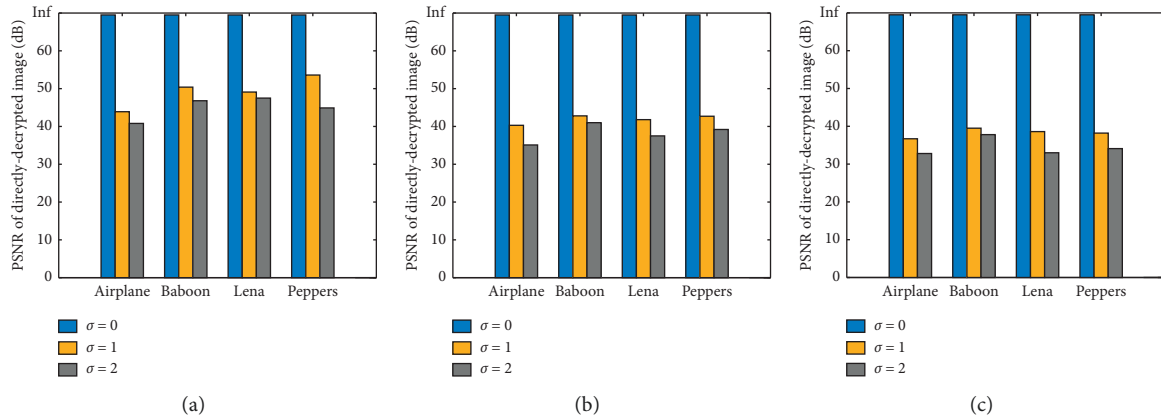


FIGURE 11: PSNR of directly decrypted image (I_d) under different values of parameter σ (dB). (a) $L = 128$. (b) $L = 256$. (c) $L = 512$.

TABLE 5: Comparisons of embedding rate between the proposed scheme with Refs. [22, 26, 40–42].

Schemes	Airplane	Baboon	Lena	Peppers
Zhang's scheme [22]	0.0500	0.0500	0.0500	0.0500
Qian and Zhang's scheme [26]	0.2952	0.2952	0.2952	0.2952
Yin et al.'s scheme [40]	0.1647	0.8767	0.2919	0.3060
Qian et al.'s scheme [41]	0.0565	0.1151	0.0559	0.0649
Wu et al.'s scheme [42]	0.0555	0.5730	0.0204	0.3420
Proposed scheme ($L = 128, \sigma = 0$)	0.6091	0.3713	0.1381	0.2388
Proposed scheme ($L = 128, \sigma = 1$)	0.8456	0.4171	0.2984	0.2891
Proposed scheme ($L = 256, \sigma = 0$)	1.3919	0.3536	0.5909	0.7957
Proposed scheme ($L = 256, \sigma = 1$)	1.6403	0.4492	0.7906	1.1590
Proposed scheme ($L = 512, \sigma = 0$)	1.8227	0.3710	0.8048	1.1459
Proposed scheme ($L = 512, \sigma = 1$)	2.1010	0.4620	1.0756	1.4823
Proposed scheme ($L = 1024, \sigma = 0$)	2.4225	0.4358	1.1024	1.6953
Proposed scheme ($L = 1024, \sigma = 1$)	2.7544	0.5371	1.4142	2.0040

5.3. *Comparison with State-of-the-Art Schemes.* Since there are few RDHEI schemes for VQ-encoded images, we chose the other five typical RDHEI schemes, i.e., Zhang's scheme [22], Qian and Zhang's scheme [26], Yin et al.'s scheme [40], Qian et al.'s scheme [41], and Wu et al.'s scheme [42], for comparing the performance of embedding rate. In detail, the RDHEI schemes [22, 26] focused on uncompressed gray scale images, the schemes [40, 41] were designed for JPEG-encoded images, and the scheme [42] was applied in palette color images. As for the proposed scheme, we utilized four kinds of VQ codebooks with sizes of 128, 256, 512, and 1024, and the parameter σ was set to 0 and 1. It was worth noting that we used the unit of bpi (bits per index) to represent the embedding rate for our RDHEI scheme of VQ-encoded images and the unit of bpp (bits per pixel) for other schemes. Comparison results for the four standard images are given in Table 5. It can be observed from the results that our scheme generally has a competitive performance of embedding rate compared with the schemes in Refs. [22, 26, 40–42].

6. Conclusions

Reversible data hiding can be used in many scenarios like security and forensics. In this work, we focus on separable reversible data hiding in encrypted VQ-encoded images, which can achieve high hiding capacity and satisfactory

image quality simultaneously. In order to protect the privacy of image contents, content-owner encrypts VQ codebook and index table with stream-cipher and permutation, respectively, and then sends the encrypted, VQ-encoded image to the data hider. In our baseline data-embedding method, the data-hider constructs index groups by grouping one high-occurrence index with one low-occurrence index; while in our optimized method, one high-occurrence index can be grouped with multiple low-occurrence indices to achieve greater hiding capacity. Through further optimizing the coefficient vector for different types of index groups, the optimal hiding capacity can be obtained by modifying the high-occurrence index into the corresponding indices in the same group according to the current to-be-embedded bits. Overall, more concentrated histogram of VQ indices leads to greater hiding capacity, and larger codebook leads to greater hiding capacity but lower directly decrypted image quality. Separable operations of data extraction, image decryption, and image recovery can be realized on the receiver side based on the availability of the encryption and data-hiding keys. Experimental results demonstrate the reversibility, security, hiding capacity, and parameter influence of our scheme, and the superiority compared with some state-of-the-art schemes. In the future work, we will further investigate the RDH for other types of encrypted data, such as video and audio.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grants 61902239, 62172280, and 62172281, in part by the Natural Science Foundation of Shanghai under Grant 21ZR1444600, in part by the Natural Science Foundation of Shandong under Grant ZR2020MF054, in part by the STCSM Capability Construction Project for Shanghai Municipal Universities under Grant 20060502300, in part by the Shandong Provincial Natural Science Foundation (ZR2019BF017), Major Scientific and Technological Innovation Projects of Shandong Province (2019JZZY010127, 2019JZZY010132, and 2019JZZY010201), and in part by Research Fund of Guangxi Key Lab of Multi-source Information Mining & Security under Grant MIMS20-03.

References

- [1] J. Jun Tian, "Reversible data embedding using a difference expansion," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 8, pp. 890–896, 2003.
- [2] Z. Zhicheng Ni, Y. Q. Yun-Qing Shi, N. Ansari, and W. Wei Su, "Reversible data hiding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 3, pp. 354–362, 2006.
- [3] D. M. Thodi and J. J. Rodriguez, "Expansion embedding techniques for reversible watermarking," *IEEE Transactions on Image Processing*, vol. 16, no. 3, pp. 721–730, 2007.
- [4] Y.-Q. Shi, X. Li, X. Zhang, H.-T. Wu, and B. Ma, "Reversible data hiding: advances in the past two decades," *IEEE Access*, vol. 4, pp. 3210–3237, 2016.
- [5] H. Yao, C. Qin, Z. Tang, and Y. Tian, "Guided filtering based color image reversible data hiding," *Journal of Visual Communication and Image Representation*, vol. 43, pp. 152–163, 2017.
- [6] F. Huang, X. Qu, H. J. Kim, and J. Huang, "Reversible data hiding in JPEG Images," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 9, pp. 1610–1621, 2016.
- [7] W. Lu, Y. J. Xue, Y. L. Yeung, H. M. Liu, J. W. Huang, and Y. Q. Shi, "Secure halftone image steganography based on pixel density transition," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 3, pp. 1137–1149, 2021.
- [8] N. M. Nasrabadi and R. A. King, "Image coding using vector quantization: a review," *IEEE Transactions on Communications*, vol. 36, no. 8, pp. 957–971, 1988.
- [9] C.-C. Chang, Y.-P. Hsieh, and C.-Y. Lin, "Lossless data embedding with high embedding capacity based on declustering for VQ-compressed codes," *IEEE Transactions on Information Forensics and Security*, vol. 2, no. 3, pp. 341–349, 2007.
- [10] J.-D. Lee, Y.-H. Chiou, and J.-M. Guo, "Reversible data hiding based on histogram modification of SMVQ indices," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 4, pp. 638–648, 2010.
- [11] C. Qin, C.-C. Chang, and Y.-C. Chen, "Efficient reversible data hiding for VQ-compressed images based on index mapping mechanism," *Signal Processing*, vol. 93, no. 9, pp. 2687–2695, 2013.
- [12] T. D. Kieu and S. Ramroach, "A reversible steganographic scheme for VQ indices based on joint neighboring coding," *Expert Systems with Applications*, vol. 42, no. 2, pp. 713–722, 2015.
- [13] C.-C. Lin, X.-L. Liu, and S.-M. Yuan, "Reversible data hiding for VQ-compressed images based on search-order coding and state-codebook mapping," *Information Sciences*, vol. 293, no. 1, pp. 314–326, 2015.
- [14] C. Qin and Y.-C. Hu, "Reversible data hiding in VQ index table with lossless coding and adaptive switching mechanism," *Signal Processing*, vol. 129, pp. 48–55, 2016.
- [15] P. Rahman and G. Dastghaibyfar, "Two reversible data hiding schemes for VQ-compressed images based on index coding," *IET Image Processing*, vol. 12, no. 7, pp. 1195–1203, 2018.
- [16] Z. Pan and L. Wang, "Novel reversible data hiding scheme for Two-stage VQ compressed images based on search-order coding," *Journal of Visual Communication and Image Representation*, vol. 50, pp. 186–198, 2018.
- [17] W. Puech, M. Chaumont, and O. Strauss, "A reversible data hiding method for encrypted images," *Proceedings of SPIE - The International Society for Optical Engineering*, vol. 6819, pp. 1–9, 2008.
- [18] C. Qin, Q. Zhou, F. Cao, J. Dong, and X. Zhang, "Flexible lossy compression for selective encrypted image with image inpainting," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 11, pp. 3341–3355, 2019.
- [19] X. Zhang, "Reversible data hiding in encrypted image," *IEEE Signal Processing Letters*, vol. 18, no. 4, pp. 255–258, 2011.
- [20] W. Hong, T.-S. Chen, and H.-Y. Wu, "An improved reversible data hiding in encrypted images using side match," *IEEE Signal Processing Letters*, vol. 19, no. 4, pp. 199–202, 2012.
- [21] X. Liao and C. Shu, "Reversible data hiding in encrypted images based on absolute mean difference of multiple neighboring pixels," *Journal of Visual Communication and Image Representation*, vol. 28, pp. 21–27, 2015.
- [22] X. Zhang, "Separable reversible data hiding in encrypted image," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 826–832, 2012.
- [23] J. Zhou, W. Sun, L. Dong, X. Liu, O. C. Au, and Y. Y. Tang, "Secure reversible image data hiding over encrypted domain via key modulation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 3, pp. 441–452, 2016.
- [24] F. Huang, J. Huang, and Y.-Q. Shi, "New framework for reversible data hiding in encrypted domain," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 12, pp. 2777–2789, 2016.
- [25] Y. Fu, P. Kong, H. Yao, Z. Tang, and C. Qin, "Effective reversible data hiding in encrypted image with adaptive encoding strategy," *Information Sciences*, vol. 494, pp. 21–36, <https://www.sciencedirect.com/science/article/abs/pii/S0020025519303561?via%3Dihub>, 2019.
- [26] Z. Qian and X. Zhang, "Reversible data hiding in encrypted images with distributed source encoding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 4, pp. 636–646, 2016.

- [27] C. Qin, W. Zhang, F. Cao, X. Zhang, and C.-C. Chang, "Separable reversible data hiding in encrypted images via adaptive embedding strategy with block selection," *Signal Processing*, vol. 153, pp. 109–122, 2018.
- [28] S. Yi and Y. Zhou, "Separable and reversible data hiding in encrypted images using parametric binary tree labeling," *IEEE Transactions on Multimedia*, vol. 21, no. 1, pp. 51–64, 2019.
- [29] K. Ma, W. Zhang, X. Zhao, N. Yu, and F. Li, "Reversible data hiding in encrypted images by reserving room before encryption," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 3, pp. 553–562, 2013.
- [30] W. Zhang, K. Ma, and N. Yu, "Reversibility improved data hiding in encrypted images," *Signal Processing*, vol. 94, no. 1, pp. 118–127, 2014.
- [31] X. Cao, L. Du, X. Wei, D. Meng, and X. Guo, "High capacity reversible data hiding in encrypted images by patch-level sparse representation," *IEEE Transactions on Cybernetics*, vol. 46, no. 5, pp. 1132–1143, 2016.
- [32] P. Puteaux and W. Puech, "An efficient MSB prediction-based method for high-capacity reversible data hiding in encrypted images," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 7, pp. 1670–1681, 2018.
- [33] Z. Yin, Y. Xiang, and X. Zhang, "Reversible data hiding in encrypted images based on multi-MSB prediction and Huffman coding," *IEEE Transactions on Multimedia*, vol. 22, no. 4, pp. 874–884, 2020.
- [34] Y. Q. Wu, Y. Z. Xiang, Y. T. Guo, J. Tang, and Z. X. Yin, "An improved reversible data hiding in encrypted images using parametric binary tree labeling," *IEEE Transactions on Multimedia*, vol. 22, no. 8, pp. 1929–1938, 2020.
- [35] Y.-C. Chen, C.-W. Shiu, and G. Horng, "Encrypted signal-based reversible data hiding with public key cryptosystem," *Journal of Visual Communication and Image Representation*, vol. 25, no. 5, pp. 1164–1170, 2014.
- [36] X. Wu, B. Chen, and J. Weng, "Reversible data hiding for encrypted signals by homomorphic encryption and signal energy transfer," *Journal of Visual Communication and Image Representation*, vol. 41, pp. 58–64, 2016.
- [37] D. Xiao, Y. Xiang, H. Zheng, and Y. Wang, "Separable reversible data hiding in encrypted image based on pixel value ordering and additive homomorphism," *Journal of Visual Communication and Image Representation*, vol. 45, pp. 1–10, 2017.
- [38] S. Xiang and X. Luo, "Reversible data hiding in homomorphic encrypted domain by mirroring ciphertext group," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 11, pp. 3099–3110, 2018.
- [39] Z. Qian, H. Zhou, X. Zhang, and W. Zhang, "Separable reversible data hiding in encrypted JPEG bitstreams," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 6, pp. 1055–1067, 2018.
- [40] Z. Yin, Y. Xiang, Z. Qian, and X. Zhang, "Unified data hiding and scrambling method for JPEG images," in *Proceedings of the 19th Pacific-Rim Conference on Multimedia*, pp. 373–383, Hefei, China, September 2018.
- [41] Z. Qian, H. Xu, X. Luo, and X. Zhang, "New framework of reversible data hiding in encrypted JPEG bitstreams," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 2, pp. 351–362, 2019.
- [42] H.-Z. Wu, Y.-Q. Shi, H.-X. Wang, and L.-N. Zhou, "Separable reversible data hiding for encrypted palette images with color partitioning and flipping verification," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 8, pp. 1620–1631, 2017.
- [43] F. Peng, Z.-X. Lin, X. Zhang, and M. Long, "Reversible data hiding in encrypted 2D vector graphics based on reversible mapping model for real numbers," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 9, pp. 2400–2411, 2019.
- [44] R. Jiang, H. Zhou, W. Zhang, and N. Yu, "Reversible data hiding in encrypted three-dimensional mesh models," *IEEE Transactions on Multimedia*, vol. 20, no. 1, pp. 55–67, 2018.
- [45] G. Schaefer and M. Stich, "Ucid – an uncompressed color image database," *Proceedings of SPIE in Storage and Retrieval Methods and Applications for Multimedia*, vol. 5307, pp. 472–480, 2004.

Research Article

An Improved Self-Training Model with Fine-Tuning Teacher/Student Exchange Strategy to Detect Computer-Generated Images

Ye Yao , Shuhui Liu, Hui Wang , Zhangyi Shen , and Xuan Ni

School of Cyberspace, Hangzhou Dianzi University, Hangzhou 310000, China

Correspondence should be addressed to Hui Wang; wanghui.ac@hotmail.com

Received 27 January 2022; Revised 25 February 2022; Accepted 11 March 2022; Published 16 April 2022

Academic Editor: Beijing Chen

Copyright © 2022 Ye Yao et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Computer-generated (CG) images have become indistinguishable from natural images due to powerful image rendering technology. Fake CG images have brought huge troubles to news media, judicial forensics, and other fields. How to detect CG image has become a key point to solve the problems mentioned above. The image classification method based on deep learning, due to its strong self-learning ability, can automatically determine the differences in the image features between CG images and natural images and can be used to detect CG images. However, deep learning often requires a large amount of labeled data, which is usually a tedious and complex task. This paper proposes an improved self-training strategy with *fine-tuning teacher/student exchange* (FTTSE) to solve the problem of missing labeled datasets. Our method is actually a strategy based on semisupervised learning to train the teacher model through labeled data and to predict the unlabeled data by the teacher model to generate pseudo labels. The student model is obtained by continuous training on the mixed dataset composed of labeled and pseudo-labeled data. A teacher/student exchange strategy is designed for iterative training; i.e., the identities of the teacher model and the student model are exchanged at the beginning of each round of iteration. And then the new teacher model is used to predict pseudo labels, and the new student model exchanged from teacher model in the previous round of iteration is fine-tuned and retrained by the mixed dataset with new pseudo labels. Furthermore, we introduced malicious image attacks to perturb the mixed dataset to improve the robustness of the student model. The experimental results show that the improved self-training model we proposed can stably maintain the image classification ability even if the testing images are maliciously attacked. After iterative training, the CG image detection accuracy of the final model increases by 5.18%. The robustness against 100% malicious attacks is also improved, where the final trained model has an accuracy improvement of 7.63% higher than the initial model. The self-training model with FTTSE strategy proposed in this paper can effectively enhance the detection ability of the existing model and can greatly improve the robustness of the model with iterative training.

1. Introduction

With the advancement of computer image rendering technology, a computer-generated (CG) image has become an important visual information carrier. Because of their unique artistry and strong sense of reality, CG images have been widely used in people's daily life and entertainment, e.g., games, virtual reality, and 3D animation. With the advancement of powerful hardware-supported rendering technology and generative adversarial network (GAN) technology, the generation for CG images has been greatly simplified, and the generated image has become more and more realistic. It is hard to distinguish the CG images from

natural images by using both human perception and computer detection. This means there are opportunities for malicious attackers to deceive facial recognition systems to impersonate others by using CG images and to create fake news to gain illegal profits, damage others' reputations, or maliciously create chaos. All the projects, such as the Digital Emily Project in 2010 [1], the Face2Face Project in 2016 [2], and the Synthesizing Obama Project in 2017 [3], prove that performing a spoofing attack has been greatly simplified, and the illegible CG images have created a focus of security concerns in the fields of news media and judiciary.

In order to solve the above mentioned problems, there are two kinds of methods proposed for CG images and

natural images classification. One is based on handcrafted features [4–10] and the other is based on convolutional neural network (CNN) [11–15]. The former usually uses feature extractors for classification. The statistical properties can be obtained from transformed images by wavelet transform or other differential operators. Now the methods based on manual statistical feature has been widely used to distinguish CG images from natural images. Rahmouni et al. [11] prove the statistical-feature-based methods perform well in image classification. CNN-based methods map the images to their corresponding labels with a function to distinguish the CG images and natural images in an end-to-end manner. Compared with the methods based on manual statistical feature, CNN-based methods have strong learning ability and can automatically learn image features. Therefore, CNN-based methods usually achieve better performance and can significantly improve the accuracy of classification. The current state-of-the-art CG image detection models are trained based on supervised learning. However, a large amount of image labels are required for training these models, which cost a lot of time for collecting correct labels. If the unlabeled images can either be used for training, the problem of insufficient labels can be effectively solved.

Current state-of-the-art CG image detection models are not as robust to changes in distribution as humans. How to quickly adapt to such changes with few labeled examples for learning is the central issue to study. As proposed by Zhu et al. [16], domain adaption is a focused research trend for transfer learning to deal with the problem. The self-training strategy proposed by Xie et al. [13] also belongs to transfer learning methods. In [16], their main contribution is to improve the adversarial robustness by using unlabeled data, and the experimental result achieves a higher accuracy with a smaller ratio between labeled and unlabeled batch size (1 : 14 and 1 : 28). The self-training strategy proposed in [13] is based on semisupervised learning (SSL). The basic idea is to find a way to use unlabeled datasets to expand labeled datasets. They firstly train the teacher model using the standard cross entropy loss with labeled data. Then the pseudo labels are generated for unlabeled images by the teacher model. The equal-or-larger student model is trained with the combined data (labeled and pseudo-labeled data) and injected noise. The student model then is used as a new teacher model for iterative training until fitting. In this self-training strategy, the model trained based on labeled data is essentially the teacher model, and the model trained based on mixed data is essentially the student model. However, there are two main drawbacks of the self-training strategy proposed in [13]. Firstly, the teacher model is directly discarded after generating the pseudo labels, which not only wastes the computing resources but also wastes a lot of internal prior knowledge of prior model training. Secondly, the student model is directly trained with mixed labels, which results in the inability to improve the model robustness.

In order to overcome the drawbacks mentioned above, we propose an improved self-training strategy with *fine-tuning teacher/student exchange* (FTTSE) to distinguish CG

images from natural images. Our contributions are summarized as follows.

- (1) The FTTSE strategy is designed by fine-tuning a previous round of the teacher model to train the subsequent student model. The main difference between our proposed strategy and [13] is the teacher/student exchange strategy. The teacher model is not directly discarded after generating the pseudo labels in our work. Except for the initial-trained teacher and student models before iteration, the previous round of teacher model is fine-tuned with learning rate decay and then it exchanges the teacher/student identity for subsequent student model training. The subsequent student models are retrained on basis of the prior knowledge learned by the previous round of teacher model.
- (2) A “Local-to-Global” strategy for pseudo-label construction is designed to reduce the interference of noisy labels in student model training. The improved pseudo-label construction strategy has strong flexibility for making classification decision no matter if by one-time prediction of the whole image or multiple-times predictions of the image blocks.
- (3) Malicious attacks are added to the mixed labeled image dataset for training student model to enhance its robustness.
- (4) A learning rate decay strategy is applied in FTTSE to prevent the model from falling into a local optimum.

The rest of this paper is organized as follows. Section 2 discusses existing related works, including the methods based on handcrafted feature, deep learning, and SSL. Section 3 illustrates the details of our proposed improved self-training strategy with FTTSE for CG image classification. In Section 4, we design a series of test experiments to verify the improved capability of the proposed strategy. This is followed by conclusion and future work in Section 5.

2. Related Works

There are two main categories of methods currently used to distinguish CG images from natural images. One is the method based on handcrafted features, which usually requires features extracted from spatial and transformed domain and uses support vector machine (SVM) as training classifier. The other is the method based on deep learning. For the handcrafted-feature-based method, Li et al. [17] proposed a multiresolution method to distinguish CG images and natural images by directly using the local binary pattern features of the image and the SVM classifier. Ng et al. [4] proposed a model based on physics-motivated features to assist in the recognition of CG images by statistically analyzing the differences between the image generation process in three aspects: object model difference, light transport difference, and image acquisition difference. In the paper, the physical characteristics, including the gamma correction in natural images and sharp structures in CG images, are described by means of the fractal geometry at the finest scale

and the differential geometry at the intermediate scale. The geometry-based SVM classifier achieved good performance in terms of both speed and accuracy. Wu et al. [18] explored the image histogram directly as the main feature for classification. The several highest bins of different image histograms are extracted as classification features to identify CG images. Although the histogram features are simple, the classifier works well in terms of detection accuracy. Lyu and Farid [19] proposed a statistical model based on the features extracted from first-order and higher-order wavelet statistics to distinguish CG images from natural images. However, the design of handcrafted features is often complex and has to be self-created for making fine distinctions. These methods generally perform well on simple dataset with images collected from limited sources, whereas they often show performance limitations when the training process is encountered with a complex dataset with images collected from many sources. To consider both global visual features and finer differences for CG image forensic, Bai et al. [20] contributed a large-scale CG images benchmark (LSCGB) with large-scale images which contain CG images with four different scenes generated by various rendering techniques and are collected with small bias on the distributions of color, brightness, tone, and saturation. They also proposed an effective texture-aware network based on the texture difference between CG and natural images to improve forensic accuracy and to exhibit the feasibility of LSCGB.

Besides the handcrafted-feature-based method, the deep-learning-based method, especially the CNN-based method, has also become a popular classification technology and has been researched with more outstanding achievements. CNN-based model is usually in an end-to-end manner, which automatically learns appropriate feature representations from superficial layer to deeper layer by existing data information. Compared with the traditional methods based on handcrafted feature which are designed and extracted from prior knowledge and assumptions, the CNN-based methods are more suitable for classification of images collected from complex source scenarios because of the powerful self-learning ability for abstraction of data features. In our work, we propose an improved teacher/student self-training strategy to detect CG images, which is essentially a semisupervised and deep-learning method. In the following subsections, the research works for image classification will be further studied in aspects of deep learning and SSL strategies, respectively.

2.1. Methods Based on Deep Learning. Benefiting from the powerful learning ability of deep learning neural network, there are many methods based on deep learning proposed to solve the problem of CG image detection. Gando et al. [21] proposed a deep convolutional neural network (DCNN) model based on fine-tuning, which includes a custom CNN-based model trained from scratch and a traditional model using handcrafted features. The fine-tuned DCNN model can automatically distinguish aggregating illustrations from photographs with detection accuracy of 96.8%. Rahmouni et al. [11] designed a special pooling layer to extract feature

statistics from complex images, which optimized the “end-to-end” CNN framework and enhanced the performance of distinguishing CG images and natural images. Yao et al. [22] proposed a CG image detection method based on sensor pattern noise and deep learning. In [22], the input images were filtered by three high-pass filters to remove low-frequency information, so as to eliminate the interference to the recognition accuracy. He et al. [23] combined CNN and recurrent neural network (RNN) to classify CG images and natural images. The authors design a dual-path neural network architecture using preprocessing operations of color space transformation and Schmid filter bank to extract image color and texture features. Exploiting the image color and texture features, the joint feature representations of local patches are learned to extract global artifact through a directed acyclic graph RNN. Capsule network was proposed in [24] and was extended in [25] to detect forged image and video for capsule-forensics applications. Tarianga et al. [26] proposed a deep convolutional recurrent model based on efficient attention.

Recently, Zhang et al. [27] used channel and pixel correlation information to reveal different features between CG images and natural images. In [27], a self-coding module was designed at the beginning of CNN and was utilized to deeply explore the correlation between the three color image channels. A new end-to-end CNN architecture called self-coding network (ScNet) was constructed with introducing hybrid correlation module and combining with existing CNN model to enhance the discrimination ability and application generality. Quan et al. [28] pointed out that the problem of blind detection of CG images is ignored in existing CNN-based methods, i.e., it is unknown whether the training images is generated by computer rendering tools or not for detection training. In order to improve the generalization ability of the model, a dual-branch neural network was designed to capture diverse features. After the normal training, the gradient information based on the CNN model is used to generate harder negative samples and then conduct enhanced training using both the original training samples and the generated negative samples. Huang et al. [29] proposed a method for effectively identifying three different kinds of digital image origin based on CNN and used a local-to-global framework to reduce training complexity. In their work, the raw pixels are used as input to CNN without the aid of “residual map.” The method behaves robustly against several common postprocessing operations.

In the latest research study, Meena and Tyagi [30] proposed a two-stream convolutional neural network to distinguish CG and photographic images. The first stream branch in the network focuses on learning different features of RGB images, while the second stream branch focuses on learning the noise features. Finally, the outcomes of two streams are merged using ensemble learning model. The experimental results show that the method maintains good performance even if the test image is processed with Gaussian noise. There are still some research works to expand the forensic functions in some special CG image application scenarios. In [31], the deep neural network was applied for image copy-move forgery localization. In [32], an

improved Xception model was applied for realistic fake face images detection. The fake face images are generated by GAN, and the experimental results show a detection accuracy improvement with the designed robust dual-stream network.

In this paper, we focus on the image classification in the application scenario of distinguishing CG images and photographic images. The network of ScNet proposed by Zhang et al. [27] adopts a distinctive mixed-channel correlation module and has strong discriminative ability and generality. Therefore, we choose ScNet as the base model for our experiments to verify the performance improvement of our proposed self-training model with FTTSE strategy.

2.2. Methods Based on SSL. Deep neural network is the state-of-the-art technology in various image classification applications, which has also obtained remarkable achievement in the research field of distinguishing CG images from natural images. However, the challenge in this research field is how to overcome the lack of labeled data to train the complex network. It is an expensive and time-consuming work to obtain a large amount of labeled data for different image classification tasks. Meanwhile, it is not feasible to manually label the data on a large scale because of the data privacy and access restrictions.

The network based on SSL proposed in [33] is one of the effective methods to solve the above problems by utilizing some labeled data and a large amount of unlabeled data for training. Mukherjee and Awadallah [34] propose an improved self-training approach that combines Bayesian deep learning and uncertainty estimation from the underlying neural network. Generally speaking, this method is a learning mechanism that uses Monte Carlo dropout as an acquisition function, selects instances from an unlabeled data pool, and uses model confidence for self-training. This method achieves excellent performance on large-scale pre-trained language models. Xie et al. [13] proposed a self-training model based on standard SSL strategy. This method firstly trains an efficient network model as teacher model on labeled images and generates pseudo labels on unlabeled images. Then it trains a higher efficient network as student model on a collection of labeled and pseudo-labeled images. The classification ability of student model is enhanced through iterative training by putting itself into the teacher model position. Compared with other SSL based models, this method improves the detection accuracy by 2.0% for ImageNet classification. Zou et al. [35] considered the noisy problem caused by the predicted pseudo labels that may result in overconfidence and wrongly placing labels on their classification in the process of self-training. In iterative training, this error bias may also propagate with iterations. To solve the problem, a self-training framework with confidence regularization was proposed. Chen et al. [36] studied an SSL model in a class-imbalanced data environment, using SSL for generating high-precision pseudo labels on minority classes. In [36], a class rebalance self-training (CRST) framework was proposed to improve existing SSL-based

methods for handling class-imbalanced data. The framework iteratively retrains the baseline SSL model to expand the sample labels by adding sample pseudo labels from the unlabeled dataset, where pseudo-labeled samples from the minority class are selected according to the estimated class distribution. They also proposed a new distribution alignment to adaptively adjust the rebalance strength, which had an outperformance comparing with other rebalancing methods based on SSL.

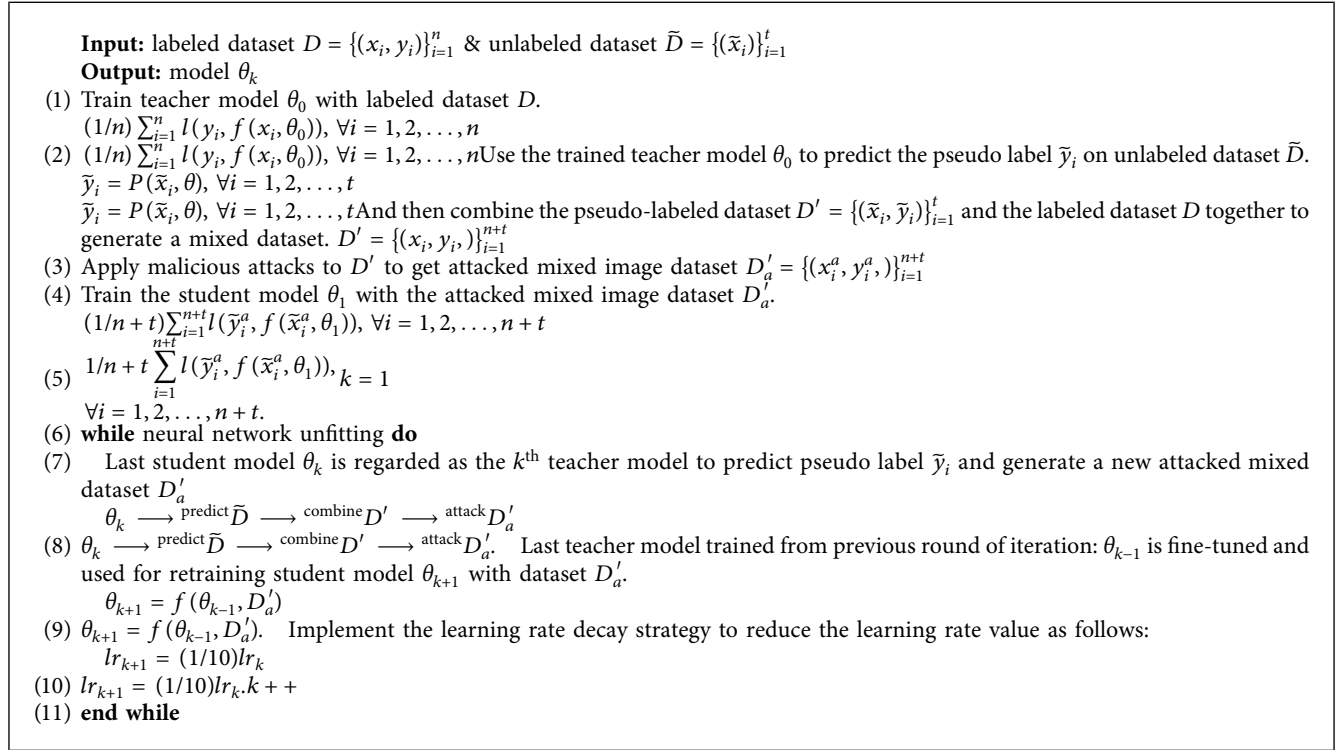
3. The Proposed Algorithm for CG Image Detection

A self-training strategy proposed by Xie et al. [13] is a method based on SSL to augment labeled datasets with unlabeled datasets. But in this method, the trained teacher model is only used to generate pseudo labels in each round of iteration and then is discarded in the next round of iteration. This results in abandoning a large amount of prior knowledge learned within the teacher model. And the computing resources consumed for training the teacher model are also wasted. In this paper, we propose an improved teacher/student exchange self-training strategy with FTTSE. The design process of the algorithm is presented in Algorithm 1. In our improved strategy, all the subsequent models are fine-tuned on the basis of the model trained in the previous round of iteration except the initial-trained teacher and student models. We design a pseudo-label construction strategy for predicting unknown image samples. In order to improve the robustness of our model, malicious attacks are applied to the images in the mixed dataset. A learning rate decay strategy is also adopted to speed up the model fitting while avoiding model falling into local optimum. Figure 1 presents the flowchart of the improved self-training model with FTTSE strategy for CG image detection.

3.1. Strategy of Teacher/Student Exchange. In this paper, we have improved the self-training strategy with teacher/student iterative training proposed by Xie et al. [13]. In the following, our proposed self-training model with FTTSE strategy will be introduced with more design details.

Before teacher/student exchange iteration, the initial teacher model θ_0 is trained from labeled data D . The teacher model θ_0 is used to predict the pseudo label \tilde{y}_i on unlabeled dataset \tilde{D} and then combine the pseudo-labeled dataset and original labeled dataset together to generate a mixed dataset D' . The student model θ_1 is trained on D'_a which is generated from D' with adding malicious attacks. Until now, the initial teacher model θ_0 and initial student model θ_1 are obtained by the first TWICE training.

Then the teacher/student exchange iteration begins. Take the first round of iteration as example. The teacher and student models firstly exchanged their identities; that is, (1) θ_1 will be used as the teacher model to generate new pseudo-label dataset; (2) θ_0 will be regarded as student model and fine-tuned for training a new student model θ_2 with new



ALGORITHM 1: Improved self-training algorithm with FTTSE.

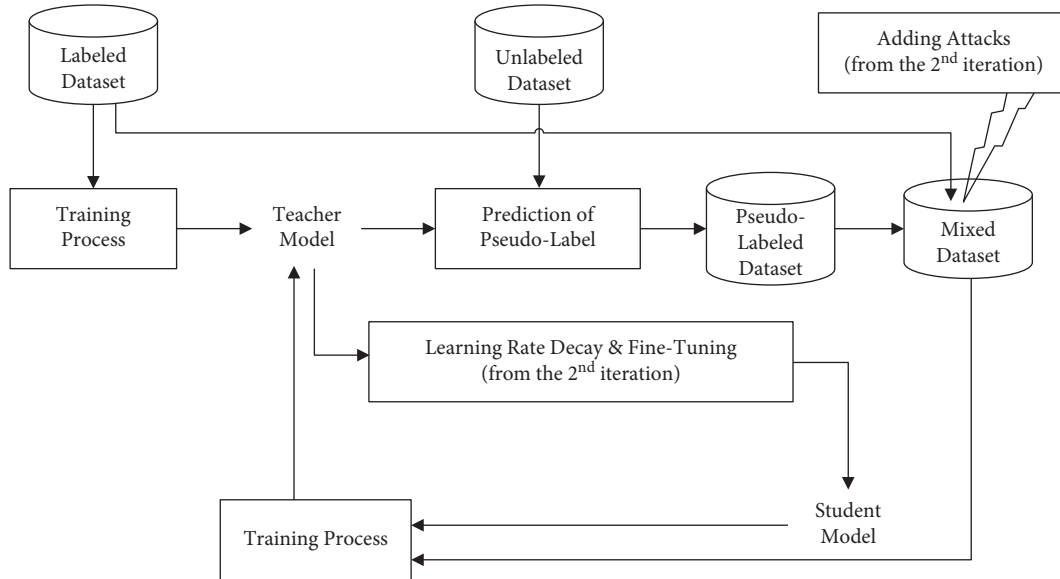


FIGURE 1: Improved self-training model with FTTSE for CG image detection.

attacked mixed dataset. At the end of each round of iteration, the learning rate will be reduced by our learning rate decay strategy. As the model θ_k continues to self-train in backward k^{th} round of iteration, the identities of teachers and students are exchanged for pseudo-label prediction and model retraining over and over again.

The improved teacher/student exchange strategy can retain the weights of the teacher model trained from the

previous round of iteration and convert them into a student model with fine-tuning to learn new image contents and distribution features. The main improvement of the proposed teacher/student exchange strategy is reflected in the generalization ability of the model. Our approach actually draws on the idea of transfer learning, which is helpful to avoid wasting of computing resources and prior knowledge and greatly speed up model training.

3.2. Strategy of Pseudo-Label Construction. In our proposed method, the pseudo label predicted by the teacher model is one kind of hard labels that conforms to a one-hot distribution. For an image x , the predicted value with teacher model θ can be obtained by function $y = P(x, \theta) = (i, i - 1)$, where $i \in [0, 1]$. In the binary classification tasks, the value of $i = 0.5$ is usually used as the cutoff, which means the image is classified as the first class when $i > 0.5$; otherwise, it is classified as the second class. In our task, i is the confidence of predicting that the image is a CG image. When $i > 0.5$, the image is predicted as a CG image, and a pseudo label is combined with the predicted image. Otherwise, the image is predicted as a natural image.

In our teacher/student training strategy, it is required for the student model to learn the prior knowledge from the teacher model, while the student model requires to surpass the teacher model in detection accuracy. When the teacher model predicts unlabeled data, there must be some prediction errors, which lead to the emergence of noisy labels. Therefore, we introduce the “Local-to-Global” strategy for pseudo-label construction. In the “Local-to-Global” strategy, the image x is randomly divided into n blocks $x_j, j \in [1, n]$, and then the teacher model is used to predict the small-sized image blocks x_j ; finally the pseudo label for image x will be decided by majority voting according to the n predicted values of x_j . The “Local-to-Global” strategy has strong flexibility for making classification decision no matter if by one-time prediction of the whole image or multiple-times predictions of the image blocks.

3.3. Malicious Attacks. In order to enhance the generalization ability and robustness of the model, kinds of malicious attacks are applied to D' to obtain a noisy dataset D'_a . Figure 2 shows an original image sample with its processed samples after seven kinds of malicious attacks, which include the following:

- (1) Noise attack: to add salt and pepper noise or Gaussian noise with $\text{SNR} \in (0.9, 1.0)$.
- (2) Translation: to move the image in a random given direction with the distance $D \in (0, 50)$.
- (3) Uniform scaling: to enlarge or shrink an image, where the scaling ratio is $r \in (0.75, 1.25)$.
- (4) Partial content blocking: a square area with size of 5×5 is randomly selected in the image to change the color to be black.
- (5) Color channel change: to convert the original RGB image to a grayscale image.
- (6) Affine transformation: a geometric transformation to transform the original vector space into another vector space by performing a linear mapping method on it.
- (7) Blurring attack: to blur the image by a Gaussian low-pass filter with kernel size $[3, 3]$ and standard deviation $\sigma = 0$.

In our strategy, after pseudo-label dataset generation by using teacher model, malicious attacks will be randomly

selected with a random parameter setting in the predefined range and be applied to partial images which are randomly selected from the mixed dataset D' with a certain probability. Here in our experimental tests, the probability of attacked image is set to be 30%. The number of images in the dataset before and after attacks remains constant, but the image content and data distribution information are expanded to enhance the generalization ability and robustness of model training.

3.4. Learning Rate Decay Strategy. Learning rate is an important hyperparameter in neural network training, which can control both the magnitude for model weights updating and the training speed. If the learning rate is too large, the weights learning will fluctuate greatly, and it is hard to get the optimal solution. If the learning rate is too small, it will result in a long training process. Therefore, learning rate setting or adjustment is crucial to neural network training. In our FTTSE strategy, a learning rate decay strategy is implemented while iterative training. Before iteration, a larger learning rate is used to speed up the initial model fitting. Then the learning rate is gradually reduced to prevent the model from falling into a local optimum in training iterations. This operation for learning rate decay is expressed mathematically as

$$lr_k = \frac{1}{10} lr_{k-1}, \quad (1)$$

where lr_k is the learning rate of k^{th} is the round of training iteration.

4. Experimental Results and Analysis

4.1. Details of Experimental Settings. In our experiment, we collected three image datasets for network training and testing, including Columbia dataset [5], DSTok dataset [37], and SPL2018 dataset [23], which are mainstream experimental datasets for current research work on CG image detection. Table 1 compares the three datasets in aspects of the number of natural images, the number of CG image, and the image size range. Besides, we will introduce the sources of image collecting in these datasets and make a simple analysis of the detection difficulty of each dataset.

The full name of Columbia dataset [5] is called Columbia Photographic Images (PIM) and Photorealistic Computer Graphics (PRCG) Dataset, which is collected by Ng et al. and is the earliest public dataset for CG image detection. The dataset contains 800 PRCG images collected from the Internet, 800 photographed PRCG images recaptured with cameras, 800 PIMs from personal collection, and 800 PIMs from Google image search. Because the image dataset has diverse image sources and the number of images for each image type is relatively small, the image classification of this dataset is relatively difficult.

DSTok dataset [37] is constructed by Tokuda et al. which contains 4850 natural images and 4850 computer-generated images. All images are collected from the Internet, including natural images with indoor and outdoor landscapes

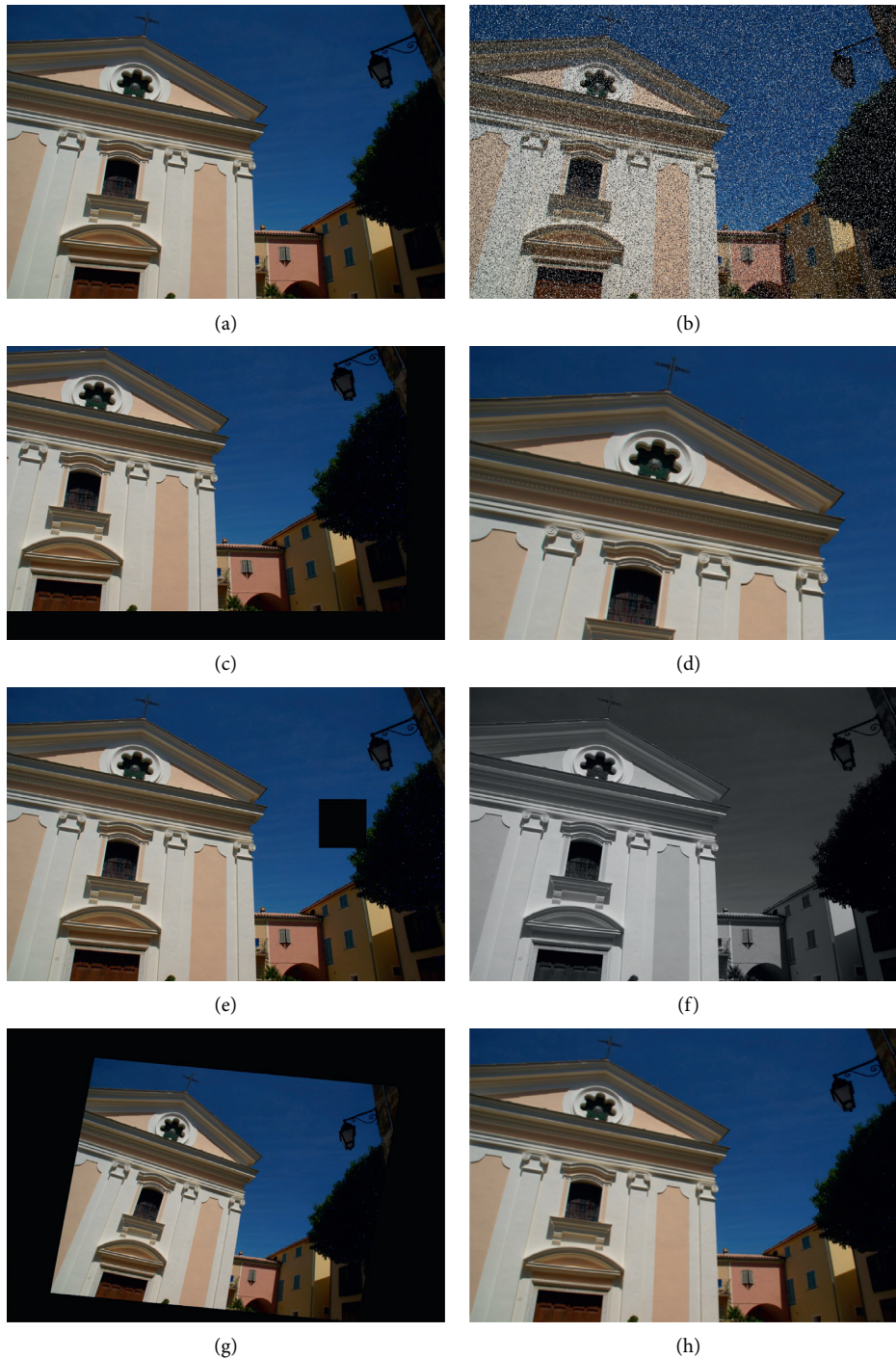


FIGURE 2: Original image sample and processed samples after different attacks. (a) No attack. (b) Noise. (c) Translation. (d) Scaling. (e) Partial content blocking. (f) Color channel change. (g) Affine transformation. (h) Blurring.

TABLE 1: Comparison of three datasets implemented in our experiment.

Dataset	Natural image number	CG image number	Image size
Columbia [5]	1600	1600	276 * 421~1398 * 1404
DSTok [37]	4850	4850	609 * 603~3507 * 2737
SPL2018 [23]	6800	6800	266 * 199~2048 * 3200

captured by various devices, and CG images collected with more content subjects, such as characters, architectures, and landscapes. DSTok dataset has a large number of images with comprehensive content categories, and it is an important dataset for CG image detection research.

SPL2018 dataset [23] is constructed by He et al. with 6800 CG images and 6800 natural images. Besides the images collected from the Internet, there are some CG images generated by more than 50 rendering software, e.g., *3DS Max*

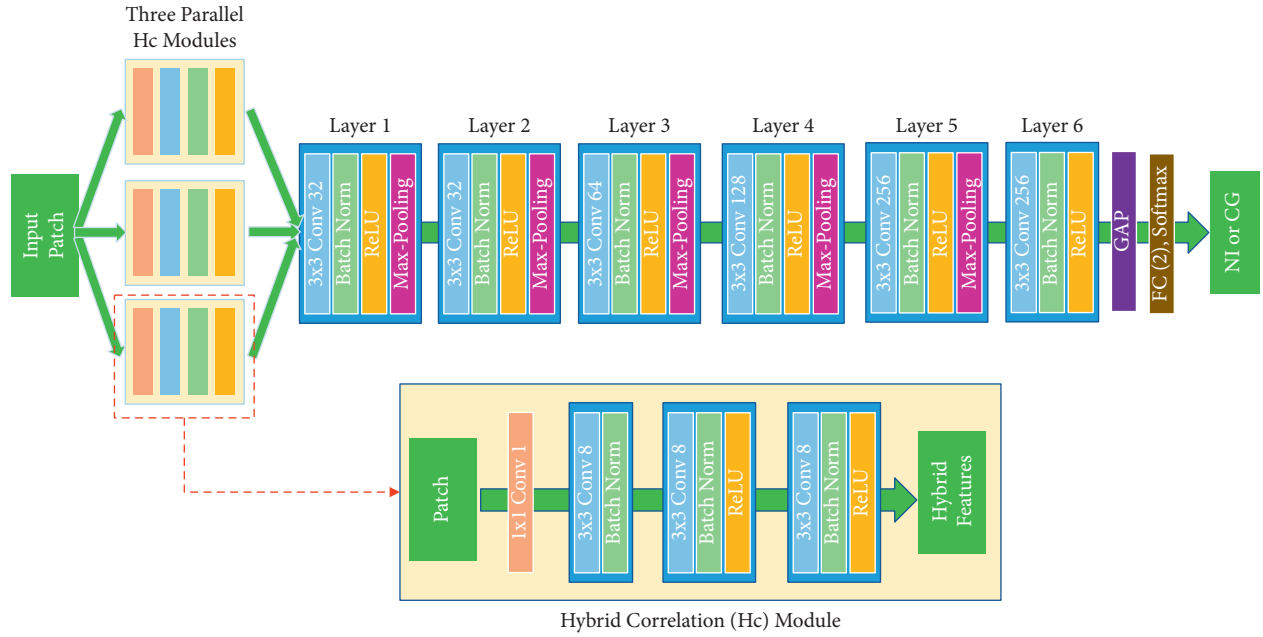


FIGURE 3: ScNet network structure proposed in [27].

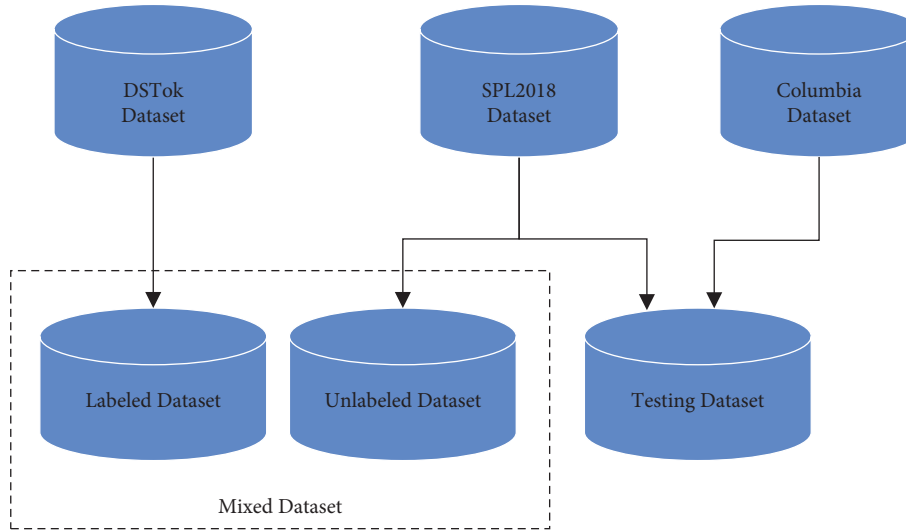


FIGURE 4: Division of different datasets.

and *Maya*. Natural images are photos captured by different types of cameras in various scenes. Besides the diverse image content subjects, the SPL2018 dataset contains images with different resolutions, especially the low-resolution images, which is more suitable to test the classification performance with more experimental scenarios requirements.

In our experiments, we use a convolutional-neural-network based model named *ScNet* proposed by Zhang et al. [27]. The key part of *ScNet* is a network structure called the self-coding module, which is efficient for deep learning of the correlation among three color channels and pixel-related features. Figure 3 shows the network structure diagram of *ScNet*. In our experiments, we compare the stability, generality, and robustness of the *ScNet* model before and after using our proposed FTTSE training strategy.

There are three kinds of dataset constructed for teacher and student model training in our work, including labeled dataset, unlabeled dataset, and testing dataset. For labeled data, we selected all images from the DSTok dataset, which is one of the most used with critical image quality in the field of CG image detection. For unlabeled dataset, we randomly selected 5000 CG images and 5000 natural images from the SPL2018 dataset and removed their labels. Then the remaining 3600 images in SPL2018 dataset and 3200 images in Columbia dataset are selected for testing dataset to verify the detection performance of the model. Figure 4 shows the dataset division process in our experiment.

For the preprocessing of image samples in the dataset, we randomly crop 20 image batches with size 224×224 for each sample and then randomly divide these cropped image

batches into training set, validation set, and testing set with a ratio of 8: 8: 1. In iterative training, we use a batch size of 32, including 16 CG images and 16 natural images. During training, the CNN parameters are optimized by stochastic gradient descent. The initial learning rate is set as 0.001, and the order of the training set is randomly shuffled after each epoch. There are four models obtained by network training at different stages of our self-training process. The first teacher model (M_0) is obtained by the first initialization training in the labeled dataset. The training process of M_0 stops after 120 epochs. The first student model (M_1) is obtained by student initialization training in the mixed dataset, which contains the labeled and the pseudo-labeled image samples. The pseudo labels are predicted by M_0 . After generating M_0 and M_1 , there are two rounds of iteration for retraining student model: M_2 and M_3 with attacked mixed dataset. The training processes of M_2 and M_3 carry out the FTTSE strategy with learning rate decay and stop after 60 epochs.

For these four models mentioned above, we designed four experiments to evaluate the model stability, benchmark the four models, validate the model generality, and evaluate the model robustness, respectively. In order to fully reflect the classification ability of the testing model, there are three experimental indicators calculated for model evaluation in terms of detection accuracy, precision, and recall. Due to the randomness of the detection results of *ScNet* model, our FTTSE self-training experiment was repeated for three times, and the average results were finally calculated to verify the network performance. All experiments are implemented on a GeForce GTX 1080Ti using the deep learning framework PyTorch0.4.1.

4.2. Stability of Training Model. In the self-training process, the model M_0 is actually similar as the model proposed by Zhang et al. [27] only with small changes in our desired experimental condition settings. Compared with the model of [27], we use the same *ScNet* network but under a different image dataset with larger size of image scene and smaller epochs of training. Following the training settings mentioned in Section 4.1, the self-training process is repeated three times in DSToK dataset to validate the network stability. For each time, the four models are evaluated by calculating their detection rates of accuracy, precision, and recall. Tables 2–4 show the experimental results of the three-times repeated self-training process, respectively. The average detection results are calculated and shown in Table 5.

As shown in Tables 2–5, the experimental results of M_0 are basically consistent with the simulation results in [27]. In our experiment, the average detection accuracy of M_0 approximately reaches 94.79%. This proves the validation of *ScNet* for CG image detection. In addition, the experimental results of M_1 , M_2 , and M_3 in the three times experiments keep a stable performance even if the training image samples are maliciously attacked. The four trained models shown in Table 3 has the best performance results among the three-times repeated experiments. Compared with M_0 , the detection accuracy of M_2 is improved by 0.58%, whereas the detection accuracy of M_3 compared with M_1 is improved by

TABLE 2: Results of stability test experiment I.

Model	Accuracy (%)	Precision (%)	Recall (%)
M_0	94.85	93.91	95.83
M_1	94.51	93.14	95.90
M_2	94.45	94.34	95.26
M_3	94.54	94.15	95.01

TABLE 3: Results of stability test experiment II.

Model	Accuracy (%)	Precision (%)	Recall (%)
M_0	94.36	92.79	95.96
M_1	94.87	93.20	95.56
M_2	94.94	93.42	96.48
M_3	95.24	93.76	96.76

TABLE 4: Results of stability test experiment III.

Model	Accuracy (%)	Precision (%)	Recall (%)
M_0	95.16	94.39	96.37
M_1	95.29	93.57	97.41
M_2	95.07	93.43	97.13
M_3	95.48	93.94	97.40

TABLE 5: Average results of three experiments.

Model	Accuracy (%)	Precision (%)	Recall (%)
M_0	94.79	93.70	96.05
M_1	94.89	93.57	96.29
M_2	94.82	93.30	96.29
M_3	95.09	93.95	96.39

0.37%. In the whole self-training process, the final training model M_3 is improved compared with the initial training model M_0 with an accuracy rate that increased by 0.88%, precision rate increased by 0.97%, and recall rate increased by 0.80%. The experimental results in Tables 2–5 reveal the stability of the model trained by our proposed FTTSE strategy and the performance improvement in CG image detection.

4.3. Benchmarking Test. Here we use the four models trained by the second experiment with the best performance as shown in Section 4.2 for benchmarking test. The remaining 3600 image samples in SPL2018 dataset with their labels, which are not used for training models, will be used as the testing set to benchmark the four models. The test results of initial training teacher model M_0 is used as the baseline for benchmarking, since there is no image sample in SPL2018 used for M_0 training. The detection accuracy, precision, and recall of the four models on remaining SPL2018 image samples are compared in Table 6. As can be seen in Table 6, the final training model M_3 performs stably higher even if there are malicious attacks applied to the training images. Due to the prior knowledge of M_0 , M_1 , and M_2 , the final model M_3 has ability of quickly learning the diagnostic features for CG image detection. Compared with the initial training teacher model M_0 , the detection accuracy of M_3 is

improved by 5.18%. The detection ability of the four models shows an upward trend in all terms of accuracy, precision, and recall. The good verification results shown in Table 6 illustrate the model improvement using FTTSE strategy.

4.4. Generality Evaluation Test. For validating the model generality, the image samples in Columbia dataset are used as unknown samples for testing detection accuracy, which were not used in any model training process. In this experiment, both 1600 natural images and 1600 CG images in the Columbia dataset are used to validate the generality of the four models to distinguish CG images from natural images.

The detection accuracy, precision, and recall of the four models on Columbia dataset are compared in Table 7. All the experimental results are visualized by bar-chart as shown in Figure 5. According to the experimental results, with the iteration of FTTSE training, the retrained model $M1$ compared with its prior fine-tuned teacher model $M0$ is improved in all terms of detection accuracy, precision, and recall. Likewise, the retrained model $M3$ is improved compared with its prior fine-tuned teacher model $M1$. The experimental results shown in Table 7 and Figure 5 validate the generalization ability of the four models, while it is also proved that the FTTSE self-training strategy can effectively strengthen the generality of model by iterative training.

4.5. Robustness Test. Since the malicious attacks are added in our iterative training process as introduced in Section 3.3, the model after implementing our teacher/student exchange strategy already has a certain robustness against various attacks. In order to sufficiently evaluate the robustness of the network model, here we enhance the attack strength or expand the subcategories of attack in our experiment. The seven kinds of malicious attack applied for robustness evaluation are reset as follows.

- (1) Noise attack with $\text{SNR} \in (0.8, 1.0)$
- (2) Translation with moving distance $D \in (0, 100)$
- (3) Uniform scaling with scaling ratio $r \in (0.5, 1.5)$
- (4) Partial content blocking with the blocking area size of 10×10
- (5) Color channel change with adding HSV color space processing
- (6) Affine transformation same as before
- (7) Blurring attack with adding media filtering with kernel size $[3, 3]$

Here we still use the four models trained from the second experiment to evaluate their robustness against strengthened attacks on the remaining 3600 image samples in SPL2018 dataset. For the testing images, we design two experiments where the strengthened attacks are randomly selected and applied to 50% and 100% of images, respectively. The experimental results are shown in Tables 8 and 9, which are visualized in Figures 6 and 7.

TABLE 6: Benchmark test results.

Model	Accuracy (%)	Precision (%)	Recall (%)
$M0$	83.42	87.76	80.74
$M1$	87.83	90.91	85.62
$M2$	88.54	91.30	86.51
$M3$	88.60	91.62	86.39

TABLE 7: Generality evaluation test results.

Model	Accuracy (%)	Precision (%)	Recall (%)
$M0$	71.09	76.76	68.95
$M1$	73.15	83.31	69.24
$M2$	72.55	80.15	69.58
$M3$	74.17	84.98	69.87

As shown in Tables 8 and 9, the detection accuracy of the initial training teacher model $M0$ only achieves 71.93% and 73.72% with 50% and 100% attacks, respectively. The detection performance of $M0$ rapidly decreased compared with the benchmark experimental results in Section 4.3 without image attacks. By introducing malicious attacks in our training strategy, the models $M1$, $M2$, and $M3$ generally perform an upward trend in the three indicators of detection accuracy, precision, and recall as shown in Figures 6 and 7. In Table 9, the final model $M3$, compared with $M0$, performs a higher detection rate with an increase of 7.63%, 6.18%, and 7.04% in terms of detection accuracy, precision, and recall. By analyzing the experimental results above, the improvement indicates our proposed FTTSE strategy can effectively enhance the robustness of the model.

4.6. Analysis of Different Dataset-Combination Settings.

In the previous experimental initialization settings, our dataset-combination setting is to select all images from the DSTok dataset as labeled data, randomly select 5000 CG images and 5000 natural images from the SPL2018 dataset as unlabeled data, and select the remaining 3600 images in SPL2018 dataset and 3200 images in Columbia dataset as testing dataset. Under this dataset-combination setting, the proposed method presents good performance in our experiments as shown in Sections 4.2–4.5. In order to verify the stability of our improved FTTSE strategy with different dataset-combination settings, we will further analyze the CG image detection accuracy of the models trained by different dataset-combination settings in this section. Here the previous dataset-combination setting is marked as “Comb1.” Besides, we add two different dataset-combination settings marked as “Comb2” and “Comb3,” respectively. In the experiment of Comb2, all the images in Columbia dataset are selected as the labeled data, 5000 CG images and 5000 natural images are randomly selected from the SPL2018 dataset as the unlabeled data, and the remaining 3600 images in SPL2018 dataset and all the images in DSTok dataset are selected as the testing dataset. In the experiment of Comb3, 5000 CG images and 5000 natural images in SPL2018 dataset are selected as the labeled data, all the images in DSTok dataset are selected as the unlabeled data, and the remaining

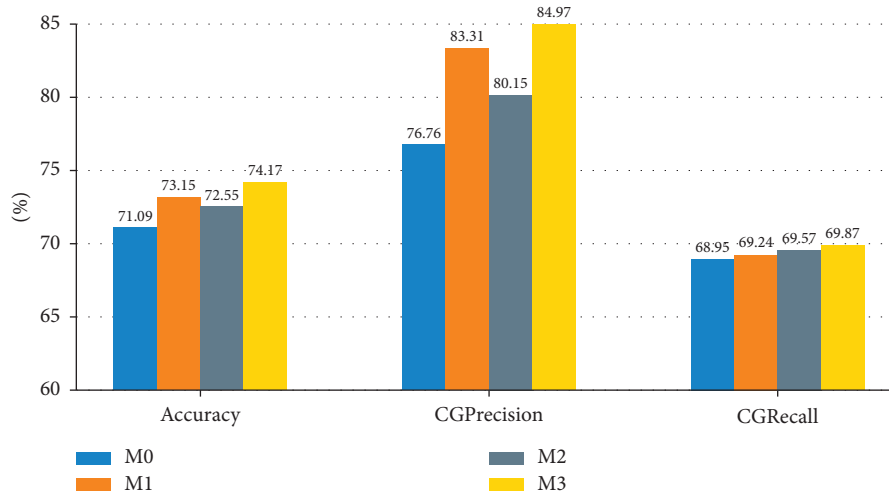


FIGURE 5: Generality evaluation test results.

TABLE 8: Robustness test results (with attack probability 50%).

Model	Accuracy (%)	Precision (%)	Recall (%)
M0	71.93	86.40	66.99
M1	84.40	91.00	80.38
M2	84.85	90.09	81.53
M3	85.12	90.84	81.51

TABLE 9: Robustness test results (with attack probability 100%).

Model	Accuracy (%)	Precision (%)	Recall (%)
M0	73.72	83.39	69.87
M1	80.98	91.02	75.78
M2	81.29	89.02	77.09
M3	81.35	89.57	76.91

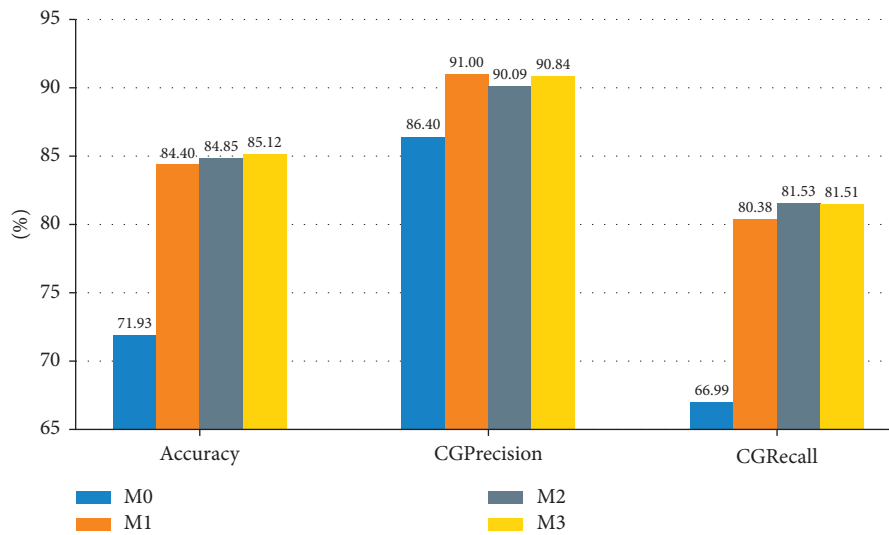


FIGURE 6: Robustness test results (with attack probability 50%).

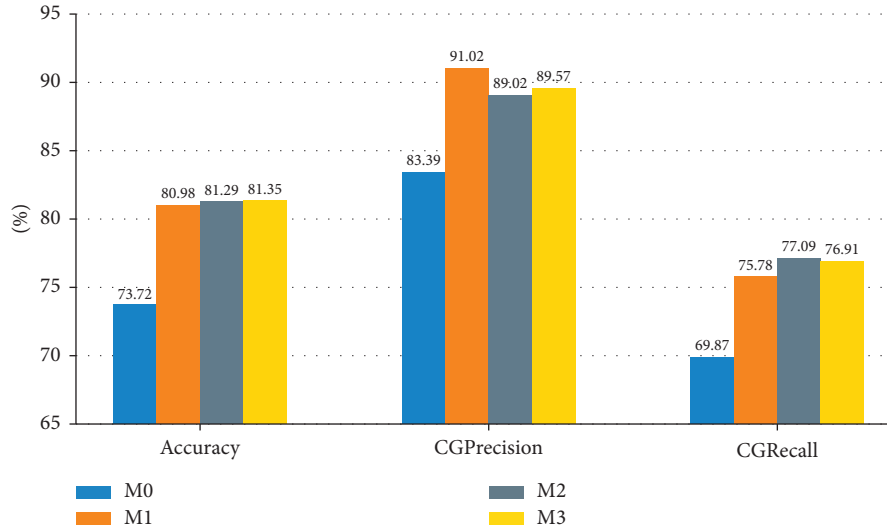


FIGURE 7: Robustness test results (with attack probability 100%).

TABLE 10: Test results for different dataset-combination settings without any attacks to the testing dataset.

Model	Accuracy (%)			Precision (%)			Recall (%)		
	Comb1	Comb2	Comb3	Comb1	Comb2	Comb3	Comb1	Comb2	Comb3
M0	83.42	81.36	95.40	87.76	79.04	94.75	80.74	82.86	96.00
M1	87.83	85.92	95.18	90.91	84.37	94.64	85.62	87.11	95.67
M2	88.54	87.85	95.16	91.30	87.33	94.91	86.51	88.24	95.39
M3	88.60	88.27	95.10	91.62	87.64	94.47	86.39	88.74	95.65

TABLE 11: Test results for different dataset-combination settings with attacks to 50% of the testing dataset.

Model	Accuracy (%)			Precision (%)			Recall (%)		
	Comb1	Comb2	Comb3	Comb1	Comb2	Comb3	Comb1	Comb2	Comb3
M0	71.93	75.44	82.13	86.40	77.44	96.54	66.99	74.45	74.93
M1	84.40	82.65	91.73	91.00	83.56	94.83	80.38	82.05	89.28
M2	84.85	85.04	91.25	90.09	86.44	94.96	81.53	84.08	88.39
M3	85.12	84.22	91.64	90.84	83.85	94.60	81.51	84.46	89.30

TABLE 12: Test results for different dataset-combination settings with attacks to 100% of the testing dataset.

Model	Accuracy (%)			Precision (%)			Recall (%)		
	Comb1	Comb2	Comb3	Comb1	Comb2	Comb3	Comb1	Comb2	Comb3
M0	73.72	70.62	70.97	83.39	76.72	97.57	69.87	67.99	63.68
M1	80.98	79.82	88.06	91.02	83.43	94.82	75.78	77.80	83.52
M2	81.29	82.34	87.19	89.02	86.03	94.92	77.09	80.10	82.20
M3	81.35	80.75	88.16	89.57	81.40	94.75	76.91	80.34	83.71

3600 images in SPL2018 dataset and all the images in Columbia dataset are selected as the testing dataset.

In each dataset-combination setting, four models are trained by the FTTSE strategy, and the detection ability of each model is benchmarked on the images in the testing dataset without any attack, with 50% malicious attack and 100% malicious attack, respectively. The attack setting is the same as introduced in Section 4.5, and the detection ability is calculated in terms of accuracy, precision, and recall,

respectively. All the experimental results for different dataset-combination settings are shown in Tables 10–12.

As shown in Table 10, the detection accuracy rates for Comb1 and Comb2 settings without any attacks are both improved from 83.42% and 81.36% to 88.60% and 88.27%. As the labeled training images and the testing images are both selected from SPL2018 in Comb3 setting, the detection accuracy performs significantly superior than that of Comb1 and Comb2 settings, and all the detection accuracy rates of

the four models keep higher than 95%. For the robustness test shown in Tables 11 and 12, it can be seen that the detection accuracy of the initial training teacher model M_0 is decreased significantly with 50% and 100% attacks, which is one of the difficult problems faced by deep learning. After the teacher/student iterative training by our FTTSE strategy, the detection accuracy of M_3 achieves 8%~13% higher than M_0 with 50% attack for testing images and 7%~17% higher than M_0 with 100% attack.

By briefly glancing at Tables 10–12, it can be seen that (1) our proposed FTTSE strategy can maintain good detection performance facing with different dataset-combination settings and (2) the proposed FTTSE strategy can enhance the robustness of the model with a significant effect in detection accuracy. In addition, it is noteworthy that in Comb2 dataset-combination setting, the number of labeled images in Columbia dataset is relatively small, but the detection capability and robustness performance with the same number of unlabeled image and the same pseudo-label construction strategy for model training can still keep stable improvement in the experiment, which further proves that our method is an effective solution to the problem of lack of labeled training samples in deep learning.

5. Conclusion

This paper proposes an improved self-training model with FTTSE strategy to distinguish CG images from naturally captured images. We improve the CG image detection accuracy of existing model through designing the new teacher/student exchange strategy, pseudo-label construction strategy, malicious-attack strategy, and learning rate decay strategy. Our experimental results show that (1) the stability of the trained model using our proposed FTTSE strategy keeps a good performance for image classification; (2) the detection accuracy of the proposed model is improved by 5.18% after iterative training; (3) the robustness of the model is improved; i.e., even if the testing image set is faced with various malicious attacks, the self-training model can still show good detection accuracy which is improved by 7.63% after iterative training.

However, during the iterative training in the experiments, the mixed training dataset will be constructed with wrong pseudo labels due to the prediction errors. The errors will be propagated and amplified with the iterative training. This causes the label distribution in mixed dataset extremely uneven in the subsequent rounds of iterative training, and the CG image detection accuracy is declining continuously. In the future work, the methods about how to overcome the imbalanced training samples in the self-training strategy and how to further improve the image classification ability will be further studied.

Data Availability

Some data and the key partial code used during the study appearing in the submitted article are available by email contacting the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

- [1] O. Alexander, M. Rogers, W. Lambeth et al., “The digital emily project: achieving a photorealistic digital actor,” *IEEE Computer Graphics and Applications*, vol. 30, no. 4, pp. 20–31, 2010.
- [2] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Niessner, “Face2face: real-time face capture and reenactment of rgb videos,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2387–2395, IEEE, Las Vegas, NV, USA, June 2016.
- [3] S. Suwajanakorn, S. M. Seitz, and I. Kemelmacher-Shlizerman, “Synthesizing Obama,” *ACM Transactions on Graphics*, vol. 36, no. 4, pp. 1–13, 2017.
- [4] T.-T. Ng, S.-F. Chang, J. Hsu, L. Xie, and M.-P. Tsui, “Physics-motivated features for distinguishing photographic images and computer graphics,” in *Proceedings of the 13th Annual ACM International Conference on Multimedia*, pp. 239–248, Singapore, November 2005.
- [5] T.-T. Ng, S.-F. Chang, J. Hsu, and P. Martin, “Columbia photographic images and photorealistic computer graphics dataset,” pp. 205–2004, Columbia University, New York, NY, USA, 2005, ADVENT Technical Report.
- [6] W. Chen, Y. Q. Shi, and G. Xuan, “Identifying computer graphics using hsv color model and statistical moments of characteristic functions,” in *Proceedings of the IEEE International Conference on Multimedia and Expo*, pp. 1123–1126, IEEE, Beijing, China, July 2007.
- [7] A. C. Gallagher and T. Chen, “Image authentication by detecting traces of demosaicing,” in *Proceedings of the 2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pp. 1–8, IEEE, Anchorage, Alaska, USA, June 2008.
- [8] R. Zhang, R.-D. Wang, and T.-T. Ng, “Distinguishing photographic images and photorealistic computer graphics using visual vocabulary on local image edges,” in *Proceedings of the International Workshop on Digital-Forensics and Watermarking*, pp. 292–305, Springer, Atlantic, NJ, USA, October 2011.
- [9] J. Wang, T. Li, Y.-Q. Shi, S. Lian, and J. Ye, “Forensics feature analysis in quaternion wavelet domain for distinguishing photographic images and computer graphics,” *Multimedia Tools and Applications*, vol. 76, no. 22, pp. 23721–23737, 2017.
- [10] E. R. S. de Rezende, G. C. S. Ruppert, A. Théophilo, E. K. Tokuda, and T. Carvalho, “Exposing computer generated images by using deep convolutional neural networks,” *Signal Processing: Image Communication*, vol. 66, pp. 113–126, 2018.
- [11] N. Rahmouni, N. Vincent, J. Yamagishi, and I. Echizen, “Distinguishing computer graphics from natural images using convolution neural networks,” in *Proceedings of the IEEE International Workshop on Information Forensics and Security (WIFS)*, pp. 1–6, IEEE, Rennes, France, December 2017.
- [12] W. Quan, K. Wang, D.-M. Yan, and X. Zhang, “Distinguishing between natural and computer-generated images using convolutional neural networks,” *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 11, pp. 2772–2787, 2018.
- [13] Q. Xie, M.-T. Luong, E. Hovy, and V. L. Quoc, “Self-training with noisy student improves imagenet classification,” in *Proceedings of the IEEE/CVF Conference on Computer Vision*

- and *Pattern Recognition*, pp. 10687–10698, Seattle, WA, USA, June 2020.
- [14] K. B. Meena and V. Tyagi, “A deep learning based method to discriminate between photorealistic computer generated images and photographic images,” in *Proceedings of the International Conference on Advances in Computing and Data Sciences (ICACDS2020): Advances in Computing and Data Sciences*, pp. 212–223, Springer, Valletta, Malta, April 2020.
- [15] K. Rajasekhar and G. Indra Sai Kumar, “Recognition of natural and computer-generated images using convolutional neural network,” in *Proceedings of the Advances in Communications, Signal Processing, and VLSI*, pp. 11–21, Springer, Hyderabad, India, April 2021.
- [16] Y. Zhu, F. Zhuang, J. Wang, J. Chen, J. Bian, and H. Xiong, “Deep subdomain adaptation network for image classification,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 4, pp. 1713–1722, 2020.
- [17] Z. Li, J. Ye, and Y. Q. Shi, “Distinguishing computer graphics from photographic images using local binary patterns,” in *Proceedings of the 2021 International Workshop on Digital-Forensics and Watermarking (IWDW2012)*, pp. 228–241, Springer, Shanghai, China, October 2013.
- [18] R. Wu, X. Li, and B. Yang, “Identifying computer generated graphics via histogram features,” in *Proceedings of the 18th IEEE International Conference on Image Processing (ICIP2011)*, pp. 1933–1936, IEEE, Brussels, Belgium, September 2011.
- [19] S. Lyu and H. Farid, “How realistic is photorealistic?” *IEEE Transactions on Signal Processing*, vol. 53, no. 2, pp. 845–850, 2005.
- [20] W. Bai, Z. Zhang, B. Li et al., “Robust texture-aware computer-generated image forensic: benchmark and algorithm,” *IEEE Transactions on Image Processing*, vol. 30, pp. 8439–8453, 2021.
- [21] G. Gando, T. Yamada, H. Sato, S. Oyama, and M. Kurihara, “Fine-tuning deep convolutional neural networks for distinguishing illustrations from photographs,” *Expert Systems with Applications*, vol. 66, pp. 295–301, 2016.
- [22] Y. Yao, W. Hu, W. Zhang, T. Wu, and Y.-Q. Shi, “Distinguishing computer-generated graphics from natural images based on sensor pattern noise and deep learning,” *Sensors*, vol. 18, no. 4, p. 1296, 2018.
- [23] P. He, X. Jiang, T. Sun, and H. Li, “Computer graphics identification combining convolutional and recurrent neural networks,” *IEEE Signal Processing Letters*, vol. 25, no. 9, pp. 1369–1373, 2018.
- [24] S. Sabour, N. Frosst, and G. E. Hinton, “Dynamic routing between capsules,” in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 3856–3866, Long Beach, CA, USA, December 2017.
- [25] H. H. Nguyen, J. Yamagishi, and I. Echizen, “Capsule-forensics: using capsule networks to detect forged images and videos,” in *Proceedings of the 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP2019)*, pp. 2307–2311, IEEE, Brighton, UK, May 2019.
- [26] D. B. Tarianga, P. Sengupta, A. Roy, R. Subhra Chakraborty, and R. Naskar, “Classification of computer generated and natural images based on efficient deep convolutional recurrent attention model,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPR)*, pp. 146–152, Las Vegas, NV, USA, July 2019.
- [27] R.-S. Zhang, W.-Z. Quan, L.-B. Fan, L.-M. Hu, and D.-M. Yan, “Distinguishing computer-generated images from natural images using channel and pixel correlation,” *Journal of Computer Science and Technology*, vol. 35, no. 3, pp. 592–602, 2020.
- [28] W. Quan, K. Wang, D.-M. Yan, X. Zhang, and D. Pellerin, “Learn with diversity and from harder samples: improving the generalization of cnn-based detection of computer-generated images,” *Forensic Science International: Digital Investigation*, vol. 35, Article ID 301023, 2020.
- [29] R. Huang, F. Fang, H. H. Nguyen, J. Yamagishi, and I. Echizen, “A method for identifying origin of digital images using a convolution neural network,” in *Proceedings of the 2020 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pp. 1293–1299, IEEE, Tokyo, Japan, December 2020.
- [30] K. B. Meena and V. Tyagi, “Distinguishing computer-generated images from photographic images using two-stream convolutional neural network,” *Applied Soft Computing*, vol. 100, Article ID 107025, 2021.
- [31] B. Chen, W. Tan, G. Coatrieux, Y. Zheng, and Y.-Q. Shi, “A serial image copy-move forgery localization scheme with source/target distinguishment,” *IEEE Transactions on Multimedia*, vol. 23, pp. 3506–3517, 2020.
- [32] B. Chen, X. Liu, Y. Zheng, G. Zhao, and Y.-Q. Shi, “A robust gan-generated face detection method based on dual-color spaces and an improved xception,” *IEEE Transactions on Circuits and Systems for Video Technology*, 2021, <https://ieeexplore.ieee.org/document/9552855>.
- [33] C. Olivier, B. Scholkopf, and Z. Alexander, “Semi-supervised learning,” *IEEE Transactions on Neural Networks*, vol. 20, no. 3, p. 542, 2006.
- [34] S. Mukherjee and A. H. Awadallah, “Uncertainty-aware self-training for text classification with few labels,” 2020, <https://arxiv.org/abs/2006.15315>.
- [35] Y. Zou, Z. Yu, X. Liu, B. V. K. Kumar, and J. Wang, “Confidence regularized self-training,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5982–5991, Seoul, Korea, October 2019.
- [36] W. Chen, K. Sohn, C. Mellina, A. Yuille, and Y. Fan, “Crest: a class-rebalancing self-training framework for imbalanced semi-supervised learning,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10857–10866, Nashville, TN, USA, June 2021.
- [37] E. Tokuda, H. Pedrini, and A. Rocha, “Computer generated images vs. digital photographs: a synergetic feature and classifier combination approach,” *Journal of Visual Communication and Image Representation*, vol. 24, no. 8, pp. 1276–1292, 2013.

Research Article

Coverless Video Steganography Based on Audio and Frame Features

Chunhu Zhang , Yun Tan , Jiaohua Qin , and Xuyu Xiang 

College of Computer Science and Information Technology, Central South University of Forestry & Technology, Changsha 410004, China

Correspondence should be addressed to Yun Tan; tantanyun@hotmail.com

Received 8 December 2021; Accepted 24 February 2022; Published 4 April 2022

Academic Editor: Beijing Chen

Copyright © 2022 Chunhu Zhang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The coverless steganography based on video has become a research hot spot recently. However, the existing schemes usually hide secret information based on the single-frame feature of video and do not take advantage of other rich features. In this work, we propose a novel coverless steganography, which makes full use of the audio and frame image features of the video. First, three features are extracted to obtain hash bit sequences, which include DWT (discrete wavelet transform) coefficients and short-term energy of audio and the SIFT (scale-invariant feature transformation) feature of frame images. Then, we build a retrieval database according to the relationship between the generated bit sequences and three features of the corresponding videos. The sender divides the secret information into segments and sends the corresponding retrieval information and carrier videos to the receiver. The receiver can use the retrieval information to recover the secret information from the carrier videos correspondingly. The experimental results show that the proposed method can achieve larger capacity, less time cost, higher hiding success rate, and stronger robustness compared with the existing coverless steganography schemes based on the video.

1. Introduction

In today's era with frequent information leakage and theft, the safe transmission of confidential information is extremely significant. Information hiding technologies can help to solve the problem of secure transmission and effective recovery of secret information. Traditional steganography schemes mainly embed secret information by changing the specific features of the carrier [1–4]. However, due to the modification of carriers, this kind of algorithms has the risk of being detected by steganalysis. The coverless steganography sets up a specific mapping relationship with the characteristics of carriers to hide secret information. It has better concealment performance than traditional information hiding algorithms since its carrier has not been changed.

The existing coverless steganography schemes are mainly divided into two categories: coverless steganography based on text [5–8] and coverless steganography based on image [9–12]. The coverless text steganography usually uses the unique features of text (such as word frequency and keywords) to hide secret information, which was first proposed

by Chen et al. [13]. They vectorized and segmented the Chinese secret information and obtained the retrieval information corresponding to the secret information from the Chinese text retrieval database to hide secret information. By using statistical features of text, Zhang et al. [14] selected the normal texts containing the secret information retrieved from the text database to hide secret information. Different from the aforementioned methods, Wang et al. [15] used the data characteristics of nonrepetitive and diverse lyrics generated by GAN to hide secret information, which had good perceptibility and embedding rate. The coverless image steganography was first proposed by Zhou in 2015 [16]. The key point of this kind of algorithms is to extract the specific features of the image, such as the texture, colour, and shape, to establish a specific mapping relationship to hide the secret information. Zhou et al. [16] divided the secret information into bit sequences and then sent the specific carrier images and auxiliary information matched with the bit sequences to the receiver. Zheng et al. [9] divided the carrier images into blocks and used the direction information of feature points of scale-invariant feature transform (SIFT) to hide the

segmented secret bit sequences. Similarly, Zhou et al. [10] used the directional gradient histogram (HOG) of non-overlapping image blocks to hide the secret bit sequences. Due to the powerful ability of deep neural network to extract features [17], some researchers introduced it to coverless steganography. Liu et al. [18] used DWT to transform images and the DenseNet network to recommend carrier images. Luo et al. [19] used the labels of the object in the carrier image to generate hash bit sequences and hid secret information by sending the carrier image containing multiple objects. The receiver used Faster RCNN [20] to extract the labels of the objects in the carrier images to recover the secret information. In addition to using the features of the image to map hash bits, some researches hid secret information based on image generation. Wu et al. [11] set up a mapping relationship between secret information and texture image and used the synthesis process of texture image to hide secret information. Chen et al. [12] divided the natural images into multiple image blocks, every one of which can represent 1-bit secret information, and retrieved the corresponding image blocks to synthesize the carrier image according to the secret information. The generative adversarial networks had caused many technological changes in the field of computer vision [21], and its ability to generate real natural images had been widely recognized. Li et al. [22] used the encoder to extract the content vector of the secret image and input it into the generative model to generate a real and natural carrier image under the penalty mechanism. Yu et al. [23] used the vectorized secret information to directly control the generative model to generate the carrier image and introduced the attention mechanism to correct the image distortion and background anomaly, so as to improve the concealment. However, when more secret information bits need to be hidden, the coverless steganography based on image or text needs to transmit a large number of carriers, which will undoubtedly arouse suspicion by external attackers and increase the risk of secret information being attacked.

Compared with image and text, there are more features that could be extracted from video to hide secret information, such as the frame image features, the temporal features between frames, and the audio features. Therefore, coverless steganography based on the video does not need to transmit too many carriers when more secret information bits need to be hidden. At the same time, due to the wide use of portable multimedia devices such as smart phones, the number of short videos is large enough on the Internet. The daily spread of video makes it an ideal covert communication carrier. These advantages provide a basis for the development of coverless steganography based on the video. However, there are a few coverless steganography methods based on the video. Tan et al. [24] calculated the directional optical flow feature of the adjacent frame images and obtained the robust histograms of oriented optical flow (RHOOFF), then mapped the hash bits according to the discrimination relationship of each component of the histogram. The optical flow information of the adjacent frame images in this scheme is sensitive to random noise, so its robustness needs to be improved. Pan et al. [25] first

performed framing processing on the video to extract valid frame images and then used the semantic information extracted from the frame images by MobilenetV2 [26] to generate hash sequences. However, this method only used a single-frame image feature of video, and its hiding capacity and hidden success rate were relatively low. At the same time, this method took a long time to train the MobilenetV2 network and the trained model also took a long time to generate a byte, which undoubtedly weakened the practicability of this scheme. Zou et al. [27] used deep neural network to extract the hash codes of frame images and set up mapping rules to improve the capacity of the scheme. However, the hash codes of the frame images were directly generated by the neural network, and the robustness of this network was poor, resulting in the weak anti-interference ability to noise.

In order to make more effective use of the features of carrier video and improve the hiding capacity and robustness, a coverless video steganography scheme based on audio and frame features is proposed in this work. First, the frame images and audio components of the carrier video are extracted. Then, three features of the two components are mapped into bit sequences, and the retrieval database is established according to the mapping relationship. The sender divides the secret information into bit sequences of equal length and then searches the retrieval information and carrier videos in the retrieval database. The retrieval information and carrier videos will be sent to the receiver. Then, the receiver can recover the secret information according to the mapping rules. The contributions of this paper are as follows:

- (1) A novel coverless video steganography scheme is proposed based on audio and frame features, which makes full use of the features of frame images and audios of the carrier videos.
- (2) The feasibility of audio features for coverless steganography is investigated, which has not been fully studied in existing researches. The short-term energy and DWT coefficient of audio are used to hide secret information. The experimental results show good performance.
- (3) The robustness, capacity, cost time, and hiding success rate are analysed and tested. The proposed method achieves good improvements compared with the existing video-based coverless steganography schemes.

The rest of the paper is arranged as follows. Preliminaries are shown in Section 2. The proposed coverless video steganography is described in Section 3. The experimental results and analysis are shown in Section 4. Finally, conclusions are drawn in Section 5.

2. Preliminaries

2.1. Short-Term Energy of Audio Signal. Since the continuous change of the audio signal with time can be characterized by a nonstationary random process and has short-term correlation, short-term analysis is generally used for audio

signal processing. The signal is divided into frames first to ensure the local stability. Then, windows are added to keep the signal continuous, as shown in Figure 1.

Assuming the audio signal is $X(n)$, the i -th frame of signal is obtained after windowing by

$$Y_i(n) = w(n) \times X((i-1) \times i_{nc} + n), 1 \leq n \leq L_f, 1 \leq i \leq f_n, \quad (1)$$

where $w(\cdot)$ is the window function with the width of w_{len} , f_n is the total number of frames, L_f is the frame length, i_{nc} is the frameshift length, and $Y_i(n)$ represents the n -th signal value of the i -th frame of the audio signal. The short-term energy can reflect the strength of the audio signal, which can be obtained by

$$E(i) = \sum_{n=0}^{L-1} Y_i^2(n), \quad 1 \leq i \leq f_n, \quad (2)$$

where L represents the length of the audio signal.

2.2. Discrete Wavelet Transform. Discrete wavelet transform (DWT) [18] is a transform method whose process is as follows:

$$\psi_{m,n}(t) = \frac{1}{\sqrt{a}} \psi\left(\frac{t-b}{a}\right) = \frac{1}{\sqrt{a_0^m}} \psi\left(\frac{t-nb_0a_0^m}{a_0^m}\right), \quad (3)$$

$$W_f(m, n) = \int_{-\infty}^{+\infty} f(t) \psi_{m,n}^*(t) dt. \quad (4)$$

Equation (3) is the discrete wavelet function of DWT, and m and n are integers; a_0 is a constant greater than 1, and b_0 is a constant greater than 0; the different values of a and b are affected by m , n , a_0 , and b_0 , and the difference of these two parameters is related to the selection of discrete wavelet function $\psi_{m,n}(t)$. Equation (4) represents the process of DWT. $f(t)$ represents the audio signal in time domain, t represents time, and $*$ represents complex conjugate value of discrete wavelet function $\psi_{m,n}(t)$.

After DWT, the audio signal can output low-frequency and high-frequency components. The low-frequency component contains the most energy of the audio signal, whereas the high-frequency component mainly contains detailed information of speech signal such as the impact of noise. As shown in Figure 2, after each DWT, the length of low-frequency information is halved, and the contour is more obvious and stable. Therefore, the low-frequency component of DWT has good stability and robustness, which can be used for information hiding.

2.3. Scale-Invariant Feature Transformation. Scale-invariant feature transformation (SIFT) has the feature of scale invariance, which is not affected by the variation of light, noise, and visual angle. Because of its excellent stability and robustness, it can be applied to information hiding [12]. The steps of SIFT feature detection are as follows:

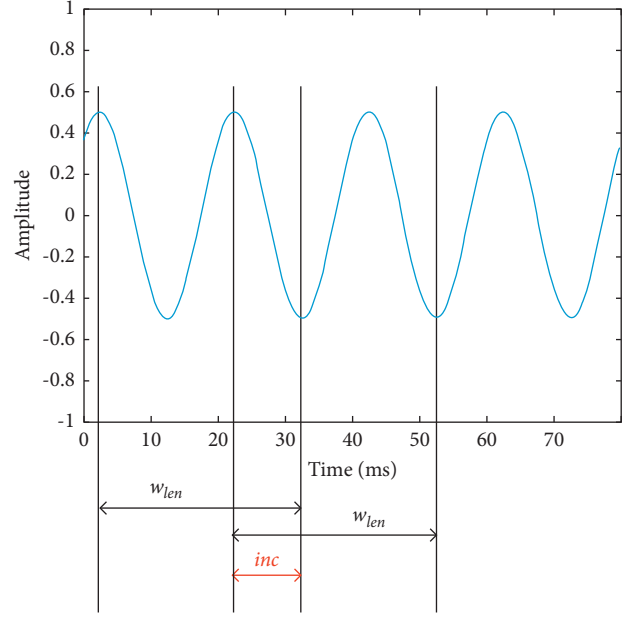


FIGURE 1: Framing process of the audio signal.

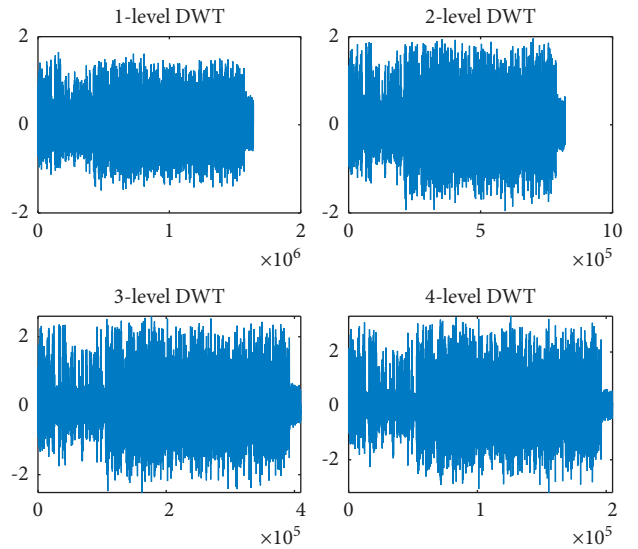


FIGURE 2: Low-frequency information after DWT.

- (1) Detect the extremum of scale space. Gaussian difference function is used to search for potential feature points with constant scale and direction for all images in scale space.
- (2) Locate the feature points. Based on the stability principle, a fine model is fitted to determine the position of the final feature points accurately.
- (3) Determine the directions of the feature points. Each feature point is assigned one or more directions based on the gradient direction of the local image. Because the image data are converted to the set direction, position, and scale features on which the

subsequent operation is also based, it provides invariance for SIFT features.

- (4) Describe the feature points. The gradient of the local image in the domain of each feature point is transformed into another representation, which could help resist a certain degree of image local distortion and illumination change.

3. Our Proposed Method

In this paper, we propose a coverless video steganography based on audio and frame features whose framework is shown in Figure 3. In our method, we first process video to get the audio and image components. Then, three features of these two components are extracted: SIFT feature, short-term energy feature, and DWT coefficient feature are used to generate hash bit sequences by different robust mapping methods. After that, the retrieval database is built according to the position information. At the sender, the secret information is divided into bit sequences, which are used to search corresponding retrieval information and videos in the retrieval database. The retrieval information and videos are sent to the receiver. After receiving the retrieval information and videos, the bit sequences are obtained by calculating the corresponding features in the video according to the retrieval information, such that the secret information can be recovered by these bit sequences.

3.1. Mapping of Bit Sequence. The mapping method of bit sequence is related to the robustness and accuracy of coverless steganography, so it is the core part of the algorithm. Our method includes three features, which are extracted from audio and image, respectively. Three mapping schemes of bit sequence are described as follows.

3.1.1. Mapping Based on Short-Term Energy of the Audio. The mapping based on short-term energy of audio is as follows:

- (1) Process the audio signal $X(n)$. According to equation (1), the audio signal $X(n)$ could be divided into frames and windowed to get f_n frames of the audio signal $Y_i(m)$. Here, we set the frame length $w_{len} = 200$ and the frameshift $i_{nc} = 80$. Then, we calculate the short-term energy of each frame audio signal according to equation (2).
- (2) Segment the energy of the audio signal. According to the principle that $L_0 = 180$ frames of total energy is used to map 1-bit information, the short-term energy $E(i)$ is segmented to get h_0 segments of short-term energy $En(h)$ according to equation (5), the first $8 \times N_0$ segments of which is used to map to generate bit sequences.

$$En(h) = \sum_{i=1}^{L_0} E(i), h_0 = \text{floor}\left(\frac{f_n}{L_0}\right)N_0 = \text{floor}\left(\frac{h_0}{8}\right), 1 \leq h \leq h_0. \quad (5)$$

- (3) Generate the hash sequences. The 8 slices of short-term energy segments are selected from $En(h)$ in sequence, and the mean value is taken as the threshold K . According to the relationship between the short-term energy and the threshold value K , we obtain the bit sequence B_1 , as shown in Figure 4. We obtain the hash sequence \bar{B}_1 by bit reversal.

$$B_1(j) = \begin{cases} 1, & \text{if } En(j) \geq K & K = \frac{(\sum_{m=1}^8 En(j))}{8} \\ 0, & \text{if } En(j) < K & 1 \leq j \leq 8 \end{cases}. \quad (6)$$

3.1.2. Mapping Based on DWT Coefficients of the Audio. We use the stable low-frequency information obtained by DWT to generate robust bit sequences as follows:

- (1) Perform DWT on the audio signal. We perform DWT on the audio signal $X(n)$ continuously three times and output the absolute value of the low-frequency information U whose length is l .
- (2) Process the coefficient of low-frequency information U . We use $L_1 = 2750$ values of low-frequency information U to map 1-bit information, and we can get $h_1 = \text{floor}(l/L_1)$ low-frequency coefficients Z_c .

$$Z_c(j) = \sum_{i=(j-1) \times L_1+1}^{j \times L_1} U(i), 1 \leq j \leq h_1. \quad (7)$$

- (3) Generate the hash sequences. By comparing the numerical relation of the adjacent DWT coefficients Z_c , we obtain the bit sequence H of length $h_1 - 1$:

$$H(j) = \begin{cases} 1, & \text{if } Z_c(j) > Z_c(j+1) \\ 0, & \text{if } Z_c(j) \leq Z_c(j+1) \end{cases} 1 \leq j \leq h_1 - 1. \quad (8)$$

- (4) Output the bit sequences in bytes. In this bit sequence H , the byte sequence B_2 is output byte by byte in sequence, as shown in Figure 5. We obtain the byte sequence \bar{B}_2 by bit inverting B_2 .
- (5) Repeat step 4 h times to output $2 \times h$ byte sequences.

$$h = \text{floor}\left(\frac{(h_0 - 1)}{8}\right). \quad (9)$$

3.1.3. Mapping Based on SIFT Feature of Frame Images. After extracting the frame images $I(m)$ from the video, the SIFT feature is used to generate a hash bit sequence from each frame image. Different from Zheng's method [12], we map the bit sequence by counting the number of SIFT feature points of image subblocks. Particularly, we use the frame image feature mapping to generate a bit sequence and then reverse it by bit to get a new bit sequence to further increase the capacity. The mapping steps are as follows:

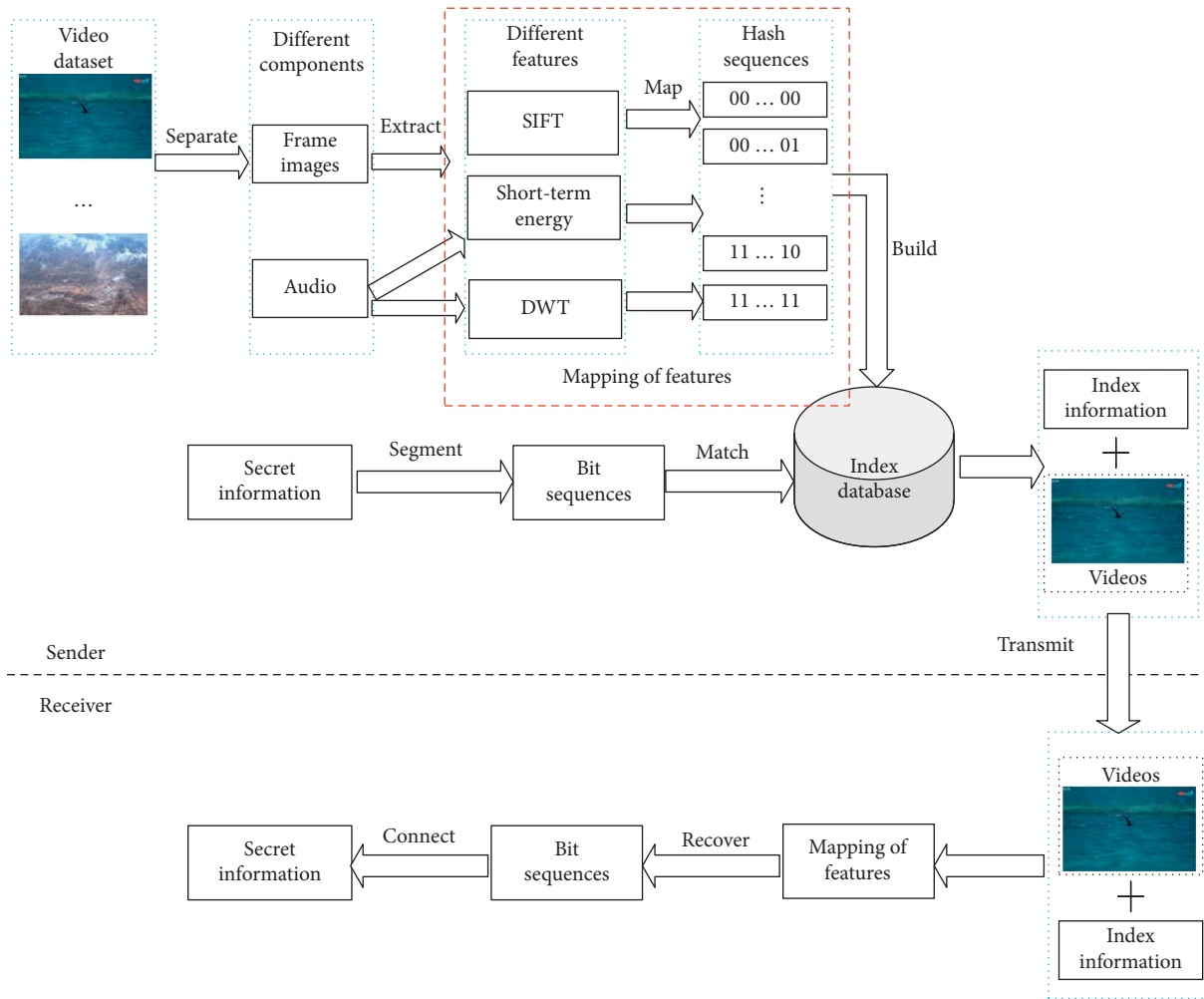


FIGURE 3: The framework of the proposed coverless video steganography scheme.

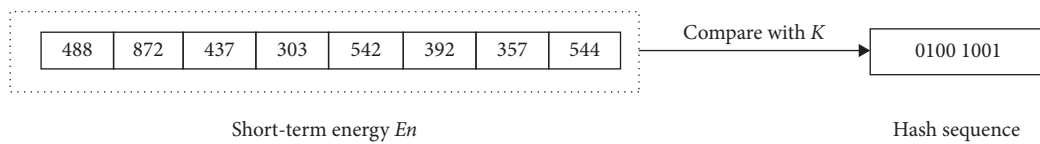


FIGURE 4: Generation of hash sequence B_1 .

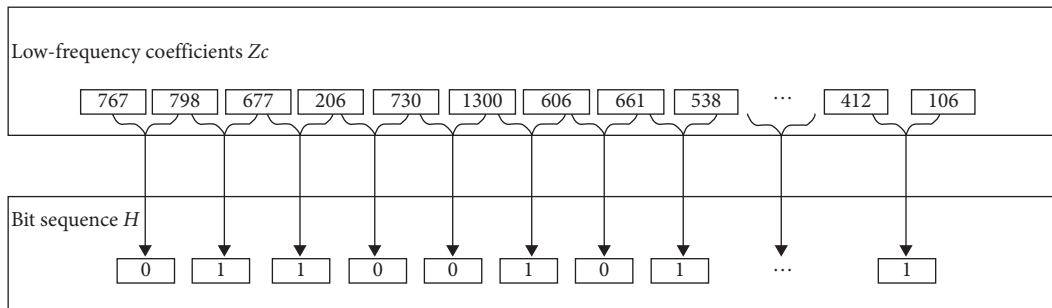


FIGURE 5: Generation of hash sequence B_2 .

- (1) Process the frame image. For a frame image, we transform it to greyscale first, uniform its size for 512×512 , and divide it into 3×3 blocks.
- (2) Generate and count SIFT feature points. We perform SFIT transformation on this frame image to obtain the location information of SIFT feature points. Then, we count the number of SIFT feature points $S(i)$ of image subblocks, $1 \leq i \leq 9$.
- (3) Generate the hash sequences. By comparing the number of SFIT feature points of different image subblocks, we obtain the hash bit sequence B_3 , as shown in Figure 6. And then, we could obtain the hash bit sequence \tilde{B}_3 by bit inverting B_3 .

$$B_3(i) = \begin{cases} 1, & \text{if } S(i) < S(i+1) \\ 0, & \text{otherwise} \end{cases}, 1 \leq i \leq 8. \quad (10)$$

- (4) Repeat steps 1, 2, and 3 until the bit sequences of all images are generated.

3.2. Establishment of Retrieval Database. The retrieval database could help sender search the carriers corresponding to the secret information, so that it is an important part of algorithm. The establishment process is as follows:

- (1) Extract two components of the video. We use Arabic numerals to mark the position of this video, which will be used to mark the video ID of the subsequent hash sequences, and then extract the frame images $I(m)$ and audio $X(n)$ from it.
- (2) Extract different features. We extract the SIFT features of the frame images $I(m)$ and mark their feature ID with 0. Then, the short-term energy features and DWT coefficient features of the audios $X(n)$ are extracted, and the feature ID is marked as 1 and 2, respectively.
- (3) Generate hash sequences and update the retrieval information. We obtain hash sequences using three mapping ways mentioned above. At the same time, we append 0 at the end of the feature ID if the hash sequence is generated by feature mapping directly, otherwise 1.
- (4) Repeat steps 1 to 3 until 256 types of different byte sequences are mapped, and the retrieval database is established, as shown in Figure 7. The algorithm of the establishment of the retrieval database is described in Algorithm 1.

It can be seen from Figure 7 that a byte sequence may have multiple corresponding retrieval information. Therefore, the sender can randomly select one of the multiple retrieval information of the byte sequence as the corresponding retrieval information, so that the same byte sequence of secret information has multiple different mapping items. It can make the auxiliary information transmitted by the sender, have more variability, increase the cracking difficulty of external attackers, and enhance the complexity of our method.

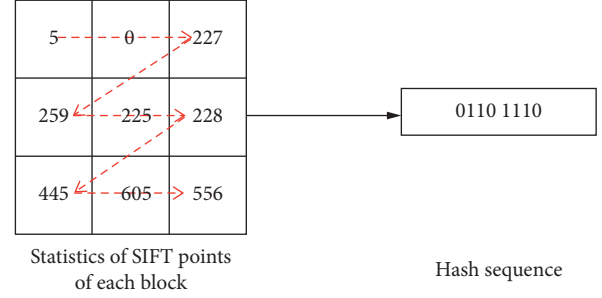


FIGURE 6: Generation of hash sequence B_3 .

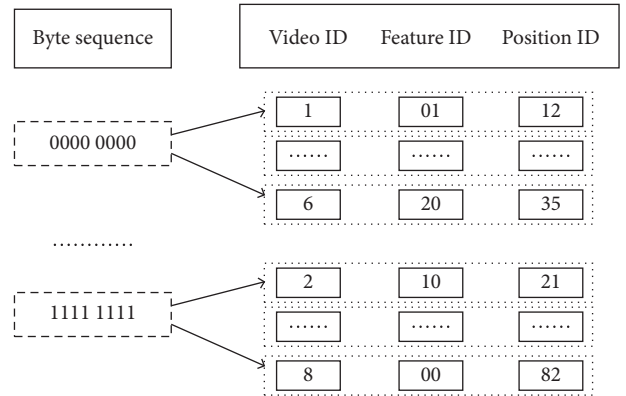


FIGURE 7: Retrieval database of videos.

3.3. Transmission of Secret Information. The specific process of secret information transmission is as follows:

- (1) Construct retrieval database of videos and obtain the carrier videos V .
- (2) For the secret information S of length L_s , segment every 8 bits (1 byte) and fills the tail with some auxiliary information to get the byte sequence $P = \{P_1, P_2, \dots, P_{L_p}\}$:

$$L_p = \begin{cases} \text{floor}\left(\frac{L_s}{8}\right) + 1, & \text{if } \text{mod}(L_s, 8) = 0, \\ \text{floor}\left(\frac{L_s}{8}\right) + 2, & \text{if } \text{mod}(L_s, 8) \neq 0. \end{cases} \quad (11)$$

If $\text{mod}(L_s, 8) \neq 0$, we pad 0 to the end of S to form a byte sequence and add 1 byte at the same time, which represents the number of 0 padded; if $\text{mod}(L_s, 8) = 0$, the sender pads a byte 0000 0000 to indicate that the original secret information has not been padded.

- (3) Get the retrieval information C_i according to P_i from the retrieval database.
- (4) Repeat step 3 until all the bytes in P have been matched to get the retrieval information $C = \{C_1, C_2, \dots, C_{L_p}\}$.
- (5) Send the retrieval information C and carrier videos V to the receiver. If both sides of the communication

have an encryption protocol, the retrieval information C can be encrypted. The algorithm of transmission of secret information is described in Algorithm 2.

3.4. Recovery of Secret Information. The specific process of secret information at receiver is as follows:

- (1) The receiver can recover the bit sequence P_i according to the retrieval information C_i and the mapping method described in Section 3.2.
- (2) Repeat step 1 until all the remaining search information of C has been matched to obtain the byte sequence $P = \{P_1, P_2, \dots, P_{L_p}\}$.
- (3) Recover secret information S according to P . If the last byte of P is 0000 0000, the last byte is directly removed to recover the secret information S . If the last byte is not 0000 0000, then according to its corresponding decimal value, the padded "0s" of the last two bytes are removed to get the original secret information S . The algorithm of secret information recovery is described in Algorithm 3.

4. Experimental Results and Analysis

The experimental environment is as follows: AMD Ryzen 5 3600 6-Core Processor CPU at 3.59 GHz, 16 Gb RAM, and NVIDIA GeForce RTX 2070 (mobile) 8G, whose driver version is 27.21.14.5671. The test software is MATLAB 2020a.

As far as we know, Pan et al. [25] proposed the first coverless video steganography method in 2020. Therefore, we will conduct performance comparison experiments with Pan's scheme on the same data set, containing some public videos we obtained from the Internet using crawler technology. The video data set consists of 240 short videos in the format of MP4. Most of these are standard definition videos, and a few are high-definition videos. Among them, the longest duration of video is about 5 minutes. The themes of this video data set include news brief, music videos, entertainment broadcast, video clip, and documentary clip. In the test of audio features, we extract the audio components of video, in which the sampling rate F_s is 44100, and remove the weak signal values—signals with an absolute value less than at the beginning and the end of the audio components to reduce noise influences. In the test experiment of the frame image features, we select some videos in the video data set for experimental testing of frame images. The partial data sets are shown in Figure 8.

Tan et al. [24] proposed a coverless steganography scheme based on optical flow analysis of video in 2021. In order to compare the latest scheme, we select the public data set UCF101 used in the paper [24] for robustness experiment comparison. UCF101 is a public video data set, the content of which is various actions and scenes. According to Tan's settings, we randomly select videos of different actions and scenes. The size of these files is about 200~800 kb and the duration of these videos is about 2~10 seconds, as shown in Figure 9.

4.1. Capacity. The hidden capacity is an important indicator of the information hiding algorithm. A coverless steganography algorithm with a large capacity can help reduce the number of carriers needed for transmission. Our scheme uses the frame image and audio components of video to establish the mapping relationship with the secret information, so the hiding capacity of our algorithm should be discussed in combination with the frame image and audio.

4.1.1. Capacity of the Audio. In this paper, we segment the audio signal and use the feature mapping of each segment of the audio signal to generate bytes. In the short-term energy feature, we first divide the audio signal into frames, then use 8×180 frames of the audio signal to map 1-byte sequence and use bit inversion to get another byte sequence. According to equation (1), the 1-second audio signal can be divided into f_n frames when the sampling rate is F_s and we can obtain

$$200 \times f_n - 80(f_n - 1) = F_s. \quad (12)$$

Then, equation (12) is transformed as

$$f_n = \frac{F_s - 80}{120}. \quad (13)$$

Therefore, a x -second video can be mapped to C_1 bytes.

$$C_1 = 2 \times \text{floor}\left(\frac{x \times f_n}{8 \times 180}\right) = 2 \times \text{floor}\left(\frac{x \times (F_s - 80)}{172800}\right). \quad (14)$$

We set $n_1 = (F_s - 80)/172800$, and then, we can simplify C_1 .

$$C_1 = 2 \times \text{floor}(x \times n_1). \quad (15)$$

In the feature of DWT coefficient, we perform four times of DWT on the audio signal to get stable low-frequency information. The length of the audio becomes $1/2^4$ of the original audio, and the value of the 1-second audio signal also changed from F_s to $F_s/2^4$ after four times of DWT transform. We use 8×2750 DWT coefficients to map 1 byte, so a video of x seconds can be mapped to generate C_2 bytes.

$$C_1 = 2 \times \text{floor}\left(\frac{x \times F_s}{16 \times 8 \times 2750}\right) = 2 \times \text{floor}\left(\frac{x \times F_s}{352000}\right). \quad (16)$$

We set $n_2 = F_s/352000$, and then, we can simplify C_2 .

$$C_2 = 2 \times \text{floor}(x \times n_2). \quad (17)$$

Therefore, the hidden capacity of the x -second audio is $C_1 + C_2$ bytes when the audio sampling rate is F_s . It can be seen from the above that the size of the audio feature capacity is related to the audio duration x , the sampling rate F_s , and the parameters L_0 and L_1 . In fact, in order to balance the robustness and capacity of the scheme, we conduct robustness tests on the values of L_0 and L_1 in Section 4.2.1 and finally determined the values of these two parameters.

4.1.2. Capacity of the Video. For a frame image, our algorithm uses frame image feature mapping and bit inversion

Input: Video database $v = \{v_1, v_2, \dots, v_a\}$
Output: Retrieval database $R = \{R_1, R_2, \dots, R_{256}\}$, Carrier videos $V = \{V_1, V_2, \dots, V_b\}$

- (1) For $i = 1$ to a
- (2) Obtain the video ID: $ID_S = i$
- (3) Extract frame images from video v_i : $I(m) = \text{ExtractImg}(v_i)$
- (4) For $j = 1$ to $\text{Length}(I(m))$
- (5) Generate hash sequence: $\text{Hash}_j = \text{CalSIFT}(I_j)$
- (6) Update retrieval database: $R = \text{update}(\text{Video ID}, \text{Feature ID}, \text{Position ID})$
- (7) End for
- (8) Extract audio from video v_i : $X(n) = \text{ExtractAud}(v_i)$
- (9) Generate hash sequence of the short-term energy features: $\text{Hash}_{\text{STE}} = \text{CalSTE}(X(n))$
- (10) For $j = 1$ to $\text{Length}(\text{Hash}_{\text{DWT}})$
- (11) Update retrieval database: $R = \text{update}(\text{Video ID}, \text{Feature ID}, \text{Position ID})$
- (12) END for
- (13) Generate hash sequence of the DWT coefficient features: $\text{Hash}_{\text{DWT}} = \text{CalDWT}(X(n))$
- (14) For $j = 1$ to $\text{Length}(\text{Hash}_{\text{DWT}})$
- (15) Update retrieval database: $R = \text{update}(\text{Video ID}, \text{Feature ID}, \text{Position ID})$
- (16) END for
- (17) $V_i = v_i$
- (18) If $\forall r \in R, r \neq \text{Null}$ then
- (19) Return R, V
- (20) End if
- (21) End for

ALGORITHM 1: Establishment of the retrieval database.

Input: Video database $v = \{v_1, v_2, \dots, v_a\}$, Secret information $S = \{S_1, S_2, \dots, S_{L_s}\}$
Output: Carrier videos $V = \{V_1, V_2, \dots, V_b\}$, Retrieval information $C = \{C_1, C_2, \dots, C_{L_p}\}$

- (1) Construct retrieval database R and obtain carrier videos V
- (2) Segment the secret information: $S' = \text{segment}(S)$
- (3) Padding the bytes sequence: $P = \text{pad}(S')$
- (4) For $i = 1$ to L_p
- (5) Search in the index information C_i corresponding to P_i
- (6) End for
- (7) Send the retrieval information C and carrier videos V to the receiver

ALGORITHM 2: Transmission of secret information.

Input: Carrier videos $V = \{V_1, V_2, \dots, V_b\}$, Retrieval information $C = \{C_1, C_2, \dots, C_{L_p}\}$
Output: Secret information $S = \{S_1, S_2, \dots, S_{L_s}\}$

- (1) Receive retrieval information C and carrier videos V .
- (2) For $i = 1$ to L_p
- (3) Obtain the byte sequence P_i according to C_i and mapping method
- (4) End for
- (5) Remove the padding bytes at the end of the byte sequence P : $S' = \text{Remove}(P)$
- (6) Connect byte sequence S' to restore secret information sequence: $S = \text{Connect}(S')$

ALGORITHM 3: Recovery of secret information.

operations to generate 2 bytes, whereas Pan's method can only map 1 byte when the robustness is optimal, but Tan's method can map 4 bytes.

For a x -second video, whose audio sampling rate is F_s and the frame images that can be extracted are M , we use the

total number of bits mapped on a certain carrier to measure the hidden capacity. The results are shown in Table 1.

It can be seen that the number of bits generated per frame image in our scheme is consistent with that of Zou's scheme, twice that of Pan's scheme, but half that of Tan's



FIGURE 8: The samples of our database.

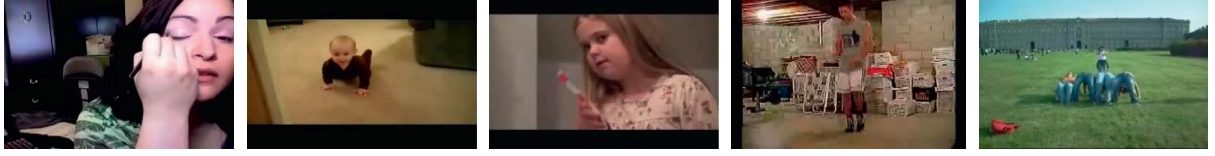


FIGURE 9: The samples of database UCF101.

TABLE 1: Capacity comparison of different methods.

Method	Capacity (bits/image)	Capacity (bits/audio)	Capacity (bits/video)
Ours	16	$8 \times (C_1 + C_2)$	$8 \times (C_1 + C_2 + 2 \times M)$
Tan's [24]	32	—	$32 \times M$
Pan's [25]	8	—	$8 \times M$
Zou's [27]	16	—	$16 \times M$

scheme. However, the other three schemes cannot use audio to map bits, but our method can map and generate $8 \times (C_1 + C_2)$ bits using the features of audio. Ideally, when the video length is long enough, the number of hash bits mapped by audio of our scheme is sufficient and the capacity of the solution can exceed the capacity of Tan's scheme.

4.2. Robustness. In the process of transmission, the carrier videos will be affected by noise or external attacks. Therefore, the robustness is an important indicator. We use external noise or attack to affect frame images and audios of the carrier video, and evaluate the robustness according to the similarities and differences between the byte sequence $P_1(t)$ recovered from the retrieval information and the byte sequence $P_0(t)$ obtained from the original secret information segmentation, which is calculated as

$$R_{\text{block}} = \frac{\sum_{t=1}^{L_p} Ac(t)}{L_p}, Ac(t) = \begin{cases} 1, & \text{if } P_0(t) = P_1(t) \\ 0, & \text{otherwise} \end{cases}, \quad (18)$$

where L_p is the byte number of sequences $P_0(t)$ and $P_1(t)$.

The paper [24] used the bit error rate to measure the robustness of the algorithm, which was calculated as

$$R_{\text{block}} = \frac{\sum_{t=1}^{L_p} Ac(t)}{L_p}, Ac(t) = \begin{cases} 1, & \text{if } P_0(t) = P_1(t) \\ 0, & \text{otherwise} \end{cases}, \quad (19)$$

where $p_0(t)$ and $p_1(t)$ represent t -th bit of sequence $P_0(t)$ and $P_1(t)$, respectively, and L_c is the total bit number.

4.2.1. Robustness Based on Audio Features. The audio is mainly affected by external Gaussian white noise. In this section, we use seven kinds of Gaussian noise under different

SNRs to test the robustness of two audio features. We test the experiment 20 times, remove the maximum and minimum values of the results, and take the average in the rest of results.

We compare the robust performance of short-term energy features with different frame numbers L_0 . Our method uses the accumulated value of 180 frames of short-term energy of the audio signal as the mapping. Table 2 shows the experimental test results of the robustness of short-term energy feature with different frame numbers L_0 . It can be seen that the robustness of short-term energy feature increases slowly with the increase of the number of frames. Moreover, the capacity will decrease if increasing L_0 . Therefore, in order to balance robustness and capacity, we set $L_0 = 180$ for better performance.

We compare the robustness performance of DWT coefficient with different segment numbers L_1 in Table 3. Our method uses the cumulative sum of 2750 segment values of U as the mapping. It can be seen that there is no big difference in the antinoise performance of DWT coefficient with different segment numbers L_1 . And there is a negative correlation between the size of L_1 and the capacity of the algorithm. In order to balance robustness and capacity, we set $L_1 = 2750$ for better performance.

We compare the robust performance of DWT coefficient feature with different times c of DWT in Table 4. It can be seen that at the beginning, with the increase of the number of DWT, the robustness is improved, but if c is greater than 4, the robustness decreases. Therefore, we set $c = 4$ for better performance.

We compare the robust performance of DWT coefficient feature with different wavelet basis functions in Table 5. It can be seen that when the wavelet basis function is rbio3.1, the comprehensive robustness of DWT coefficient is better; therefore, we set the wavelet basis function as rbio3.1.

TABLE 2: R_{block} of short-term energy with different frame numbers L_0 .

L_0	Gauss noise						
	snr = 0	snr = 3	snr = 5	snr = 10	snr = 15	snr = 20	snr = 30
180	0.8607	0.9070	0.9276	0.9600	0.9775	0.9867	0.9952
360	0.8894	0.9264	0.9420	0.9674	0.9810	0.9895	0.9964
540	0.9010	0.9325	0.9493	0.9721	0.9845	0.9908	0.9968
720	0.9150	0.9436	0.9568	0.9772	0.9875	0.9924	0.9975
900	0.9154	0.9484	0.9584	0.9783	0.9882	0.9935	0.9983
1080	0.9187	0.9478	0.9592	0.9793	0.9886	0.9939	0.9984
1260	0.9229	0.9515	0.9615	0.9802	0.9891	0.9936	0.9981
1440	0.9288	0.9511	0.9640	0.9802	0.9880	0.9922	0.9979
1620	0.9323	0.9539	0.9650	0.9805	0.9891	0.9938	0.9978
1800	0.9335	0.9557	0.9664	0.9824	0.9906	0.9945	0.9982

TABLE 3: R_{block} of short-term energy with different segment numbers L_1 .

L_1	Gauss noise						
	snr = 0	snr = 3	snr = 5	snr = 10	snr = 15	snr = 20	snr = 30
1375	0.8334	0.8773	0.9015	0.9423	0.9659	0.9800	0.9923
2750	0.8688	0.9061	0.9238	0.9567	0.9743	0.9849	0.9942
5500	0.8803	0.9160	0.9341	0.9654	0.9811	0.9898	0.9958
8250	0.8880	0.9208	0.9379	0.9663	0.9813	0.9882	0.9953
11000	0.8853	0.9211	0.9350	0.9667	0.9816	0.9905	0.9967

TABLE 4: R_{block} of DWT coefficient with different times c .

c	Gauss noise						
	snr = 0	snr = 3	snr = 5	snr = 10	snr = 15	snr = 20	snr = 30
1	0.8053	0.8606	0.8875	0.9390	0.9684	0.9818	0.9940
2	0.8397	0.8846	0.9105	0.9503	0.9727	0.9853	0.9953
3	0.8565	0.8980	0.9194	0.9559	0.9758	0.9864	0.9956
4	0.8688	0.9061	0.9238	0.9567	0.9743	0.9849	0.9942
5	0.8646	0.9024	0.9225	0.9547	0.9738	0.9845	0.9948
6	0.8351	0.8791	0.9021	0.9445	0.9678	0.9815	0.9935
7	0.7294	0.7963	0.8314	0.8990	0.9404	0.9644	0.9875

TABLE 5: R_{block} of DWT coefficient with different wavelet basis functions.

Wavelet	Gauss noise						
	snr = 0	snr = 3	snr = 5	snr = 10	snr = 15	snr = 20	snr = 30
db5	0.8574	0.8975	0.9182	0.9541	0.9749	0.9864	0.9953
db15	0.8527	0.8925	0.9132	0.9518	0.9724	0.9848	0.9946
coif1	0.8569	0.8982	0.9181	0.9541	0.9730	0.9850	0.9949
coif5	0.8540	0.8960	0.9148	0.9511	0.9714	0.9832	0.9935
fk4	0.8555	0.8956	0.9167	0.9510	0.9718	0.9835	0.9938
fk18	0.8547	0.8954	0.9162	0.9515	0.9726	0.9842	0.9947
sym2	0.8556	0.8957	0.9167	0.9519	0.9729	0.9843	0.9947
sym8	0.8578	0.8989	0.9168	0.9532	0.9726	0.9847	0.9944
dmey	0.8547	0.8970	0.9184	0.9535	0.9732	0.9841	0.9946
bior1.1	0.8498	0.8925	0.9149	0.9525	0.9710	0.9830	0.9941
bior3.1	0.5131	0.6424	0.7118	0.8323	0.9048	0.9465	0.9827
rbio3.1	0.8697	0.9094	0.9270	0.9574	0.9758	0.9859	0.9951
rbio3.3	0.8709	0.9053	0.9252	0.9559	0.9752	0.9854	0.9949
rbio3.5	0.8683	0.9054	0.9230	0.9554	0.9737	0.9854	0.9942
rbio4.4	0.8587	0.8992	0.9190	0.9545	0.9749	0.9858	0.9948
rbio5.5	0.8452	0.8895	0.9117	0.9496	0.9713	0.9834	0.9953
rbio6.8	0.8600	0.8986	0.9206	0.9555	0.9754	0.9854	0.9948

TABLE 6: R_{block} of two audio features.

Gauss noise	snr = 0	snr = 3	snr = 5	snr = 10	snr = 15	snr = 20	snr = 30
Robustness	0.8624	0.9068	0.9270	0.9599	0.9768	0.9867	0.9950

TABLE 7: R_{block} of different methods on our database.

Attack	Parameter	Pan's [25]	Zou's [27]	Ours
Salt and pepper noise	$\sigma = 0.001$	0.7335	0.7323	0.8765
	$\sigma = 0.005$	0.4776	0.4252	0.7595
Gauss noise	$\sigma = 0.001$	0.2269	0.5739	0.6864
	$\sigma = 0.005$	0.2267	0.2374	0.5104
	$\sigma = 0.01$	0.2250	0.1330	0.4252
Speckle noise	$\sigma = 0.01$	0.4291	0.5151	0.6224
	$\sigma = 0.05$	0.2949	0.2070	0.4162
JPEG compression	Q = 10	0.0886	0.1752	0.5990
	Q = 70	0.5863	0.8030	0.8029
	Q = 90	0.7221	0.8897	0.8601
Centred cropping	Ratio = 10%	0.2267	0.0467	0.3672
	Ratio = 20%	0.0853	0.0213	0.1512
Edge cropping	Ratio = 10%	0.2436	0.1909	0.7410
	Ratio = 20%	0.1472	0.0877	0.5730
Rotation	Rotation angle = 10°	0.0333	0.0131	0.2351
	Rotation angle = 15°	0.0333	0.0067	0.1551
Translation	(16, 10)	0.1741	0.1665	0.5820
	(40, 25)	0.0557	0.0627	0.3238
Mean filtering	Window size: 3 × 3	0.5958	0.5822	0.6155
	Window size: 5 × 5	0.3542	0.2788	0.4847
Gamma correction	Factor = 0.8	0.4395	0.4579	0.7271
Colour histogram equalization	None	0.0906	0.0465	0.3441

The overall robustness performance of the two audio features is shown in Table 6. We set the short-term energy feature parameter as $L_0 = 180$, the DWT coefficient feature parameter as $L_1 = 2750$, the wavelet basis function as `rbio3.1`, and the number of DWT as $c = 4$. It can be seen that the robustness of the audio feature mapping method is strong, which can reach 86% under the Gaussian noise with SNR of 0.

4.2.2. Robustness Based on Frame Image Features. In this section, we use a variety of geometric attacks and noise attacks with different parameters to test the robustness of different methods on our video data set. According to the setting of paper [25], the j of Pan's scheme is set to 9. The experimental results of different methods on our database are shown in Table 7.

It can be seen that the robust performance of our method is better than that of Zou's and Pan's method under most external image attacks. Because these two schemes use the neural network to directly extract the features of the frame images, the influence of the pixel values has a great impact on the output results of the network, resulting in the weak anti-interference ability to noise. In particular, the pixel matrix of the frame image will be quite different from the original matrix if the video encounters geometric attack. Affected by the prior knowledge of the training set, the extracted features of the neural network may be quite

different from the original features. With the scale invariance of SIFT, our method can stably extract the feature points of the frame images and has good robustness to noise attack and geometric attack.

In order to compare the experiment with the state-of-the-art scheme [24], we use equation (19) to compare the robustness experiments with Pan's scheme and Tan's scheme on the data set UCF101. According to the setting of paper [24], the bin number N is set to 8, and the subblock number S is set to 4; according to the setting of paper [25], j is set to 9. The results are shown in Table 8. We can see that most of the experimental data of our scheme is stronger than the other two schemes, especially the anticompensation performance. Compared with other antinoise performance, anticompensation performance is particularly important for coverless steganography based on video. Because carrier video generally undergoes a compression step before sending, which often damages the video content.

4.3. Efficiency Analysis. The complexity and efficiency will affect the feasibility and practicability of the steganography scheme. The cost of our scheme is mainly related to the map of hash bits and three features, because it involves the calculation and mapping of three features. We measure the efficiency of the schemes based on the time

TABLE 8: R_{bit} of different methods on database UCF101.

Attack	Parameter	Tan's [24]	Pan's [25]	Ours
Salt and pepper noise	$\sigma = 0.001$	0.9986	0.9559	0.9763
	$\sigma = 0.005$	0.9923	0.9063	0.9504
	$\sigma = 0.01$	0.9877	0.8731	0.9186
Gauss noise	$\sigma = 0.001$	0.7005	0.7889	0.9139
	$\sigma = 0.005$	0.6485	0.7889	0.8292
	$\sigma = 0.01$	0.6198	0.7821	0.7473
Speckle noise	$\sigma = 0.001$	0.8235	0.9150	0.9466
	$\sigma = 0.005$	0.8098	0.8698	0.8829
	$\sigma = 0.01$	0.8000	0.8431	0.7952
Compressed MPEG-4 file with H.264 (.mp4 file)	None	0.9589	0.9172	0.9594
Compressed motion JPEG 2000 file (.mj2 file)	None	0.8476	0.9676	0.9790

TABLE 9: Time cost comparison of different methods.

Method	Tan's [24]	Pan's [25]	Zou's [27]	Ours
Time cost	0.7416 s/B	1.3769 s/B	1.2994 s/B	0.1755 s/B

required to hide a byte, and the unit is "s/B." From the results in Table 9, it can be seen that the time required in our scheme is the least, which is about one quarter of the time cost of Tan's method and about one seventh of the time cost of Zou's and Pan's methods. Therefore, the cost of our scheme is the lowest, which undoubtedly enhances the feasibility of our scheme.

4.4. Hiding Success Rate. Information hiding algorithm should not only consider the capacity of the method but also pay attention to the hiding success rate, which can be expressed by the number of different bytes that a video can hide. Hiding success rate can reflect the effectiveness of the algorithm, and its calculation formula is shown in

$$S = \frac{Q}{2^w}, \quad (20)$$

where Q is the total number of bit sequences generated by multiple videos and $w = 8$ in this experiment.

We use 85 videos in the video data set to test our method and Pan's method, and the results are shown in Figure 10. The hiding success rate of our method is always higher than that of Pan's method, and only 9 videos are enough to map 256 types of different bit sequences. This is because we use three features and bit inversion operation, and thus, a video can generate a variety of hash sequences. The hiding success rate of Pan's method can only approach 99% with 85 videos, which means that the redundancy of bit sequences generated by multiple videos is high and a large number of videos are needed to map all kinds of bit sequences.

4.5. Security Analysis. The coverless video steganography based on audio and frame features proposed in this paper has multiple securities as follows:

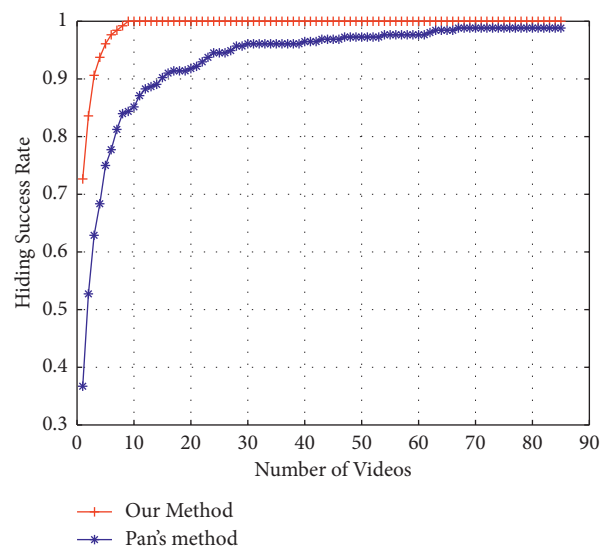


FIGURE 10: Comparison of hiding success rate.

- (1) We use three features of video to map the hash bit sequences and hide the secret information, rather than modifying the carrier video. Therefore, this method could resist steganalysis tools, which could ensure the security of secret information.
- (2) The carrier video used by our method is from the abundant short videos on the Internet, which could greatly reduce the attention of the outside world to the secret communication, so as to improve the security of communication.

5. Conclusion

A coverless video steganography based on audio and frame features is proposed in this work, which makes full use of short-term energy feature, DWT coefficient feature, and SIFT feature of video to map hash bit sequences and hide secret information. The experimental results show that, compared with the existing coverless video steganography, our method has larger capacity, less time cost, higher success rate of hiding, and stronger robustness to most external

attacks. In the future, we will try to further improve the robustness and capacity.

Data Availability

The video database we built can be obtained upon request to the corresponding author. The UCF101 data used to support the findings of this study are available at <https://www.crcv.ucf.edu/data/UCF101.php>.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (Grant 62002392), the National Natural Science Foundation of Hunan (Grants 2020JJ4140 and 2020JJ4141), the Science Research Projects of Hunan Provincial Education Department (Grant 19B584), the Degree & Postgraduate Education Reform Project of Hunan Province (Grant 2019JGYB154), and the Postgraduate Excellent teaching team Project of Hunan Province (Grant [2019] 370-133).

References

- [1] B. Wang, W. Kong, N. Li, N. Neal, W. Xiong, and N. N. Xiong, "A dual-chaining watermark scheme for data integrity protection in Internet of things," *Computers, Materials & Continua*, vol. 58, no. 3, pp. 679–695, 2019.
- [2] D. R. Vinay and B. J. Ananda, "A novel secure data hiding technique into video sequences using RVIHS," *International Journal of Computer Network and Information Security*, vol. 13, no. 2, pp. 53–65, 2021.
- [3] X. Zhong, P. C. Huang, S. Mastorakis, and F. Y. Shih, "An automated and robust image watermarking scheme based on deep neural networks," *IEEE Transactions on Multimedia*, vol. 23, pp. 1951–1961, 2020.
- [4] S. Ren, T. Zhang, M. Wang, and K. Shahzad, "Identifiable tampering multi-carrier image information hiding algorithm based on compressed sensing," *IEEE Access*, vol. 8, Article ID 214992, 2020.
- [5] X. Chen and S. Chen, "Text coverless information hiding based on compound and selection of words," *Soft Computing*, vol. 23, no. 15, pp. 6323–6330, 2019.
- [6] X. Chen, S. Chen, and Y. Wu, "Coverless information hiding method based on the Chinese character encoding," *Journal of Internet Technology*, vol. 18, no. 2, pp. 313–320, 2017.
- [7] Z. Zhou, Y. Mu, N. Zhao, Q. M. J. Wu, and C.N. Yang, "Coverless information hiding method based on multi-keywords," in *Proceedings of the International Conference on Cloud Computing and Security*, pp. 39–47, Nanjing, China, July, 2016.
- [8] Z. Zhou, Y. Mu, C.N. Yang, and N. Zhao, "Coverless multi-keywords information hiding method based on text," *International Journal of Security and Its Applications*, vol. 10, no. 9, pp. 309–320, 2016.
- [9] S. Zheng, L. Wang, B. Ling, and D. Hu, "Coverless information hiding based on robust image hashing," in *Proceedings of the International Conference on Intelligent Computing*, pp. 536–547, Nanjing, China, July, 2017.
- [10] Z. Zhou, Q. J. Wu, C.-N. Yang, X. Sun, and Z. Pan, "Coverless image steganography using histograms of oriented gradients-based hashing algorithm," *Journal of Internet Technology*, vol. 18, no. 5, pp. 1177–1184, 2017.
- [11] K. Wu and C. Wang, "Steganography using reversible texture synthesis," *IEEE Transactions on Image Processing*, vol. 24, no. 1, pp. 130–139, 2014.
- [12] X. Y. Chen, A. Q. Qiu, X. M. Sun, S. Wang, and G. Wei, "A high-capacity coverless image steganography method based on double-level index and block matching," *Mathematical Biosciences and Engineering: MBE*, vol. 16, no. 5, pp. 4708–4722, 2019.
- [13] X. Chen, H. Sun, Y. Tobe, Z. Zhou, and X. Sun, "Coverless information hiding method based on the Chinese mathematical expression," in *Proceedings of the International Conference On Cloud Computing And Security*, pp. 133–143, Nanjing, China, August, 2015.
- [14] J. Zhang, H. Huang, L. Wang, H. Lin, and D. Gao, "Coverless text information hiding method using the frequent words hash," *International Journal on Network Security*, vol. 19, no. 6, pp. 1016–1023, 2017.
- [15] C. Wang, Y. Liu, Y. Tong, and J. Wang, "GAN-GLS: generative lyric steganography based on generative adversarial networks," *Computers, Materials & Continua*, vol. 69, no. 1, pp. 1375–1390, 2021.
- [16] Z. Zhou, H. Sun, R. Harit, X. Chen, and X. Sun, "Coverless image steganography without embedding," in *Cloud Computing and Security*, pp. 123–132, Springer, Switzerland, 2015.
- [17] B. Chen, W. Tan, G. Coatrieux, Y. Zheng, and Y.Q. Shi, "A serial image copy-move forgery localization scheme with source/target distinguishment," *IEEE Transactions on Multimedia*, vol. 23, pp. 3506–3517, 2021.
- [18] Q. Liu, X. Xiang, J. Qin, Y. Tan, J. Tan, and Y. Luo, "Coverless steganography based on image retrieval of DenseNet features and DWT sequence mapping," *Knowledge-Based Systems*, vol. 192, no. 1, Article ID 105375, 2020.
- [19] Y. Luo, J. Qin, X. Xiang, and Y. Tan, "Coverless image steganography based on multi-object recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 7, pp. 2779–2791, 2021.
- [20] S. Ren, K. He, R. Girshick, and J. Sun, "Towards real-time object detection with region proposal networks," *Advances in Neural Information Processing Systems*, vol. 28, no. 1, pp. 91–99, 2015.
- [21] B. Chen, X. Liu, Y. Zheng, G. Zhao, and Y.Q. Shi, "A robust GAN-generated face detection method based on dual-color spaces and an improved Xception," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 99, p. 1, 2021.
- [22] Q. Li, X. Wang, X. Wang, and Y. Shi, "CCCIH: content-consistency coverless information hiding method based on generative models," *Neural Processing Letters*, vol. 53, no. 6, pp. 4037–4046, 2021.
- [23] C. Yu, D. Hu, S. Zheng, W. Jiang, M. Li, and Z.Q. Zhao, "An improved steganography without embedding based on attention GAN," *Peer-to-Peer Networking and Applications*, vol. 14, no. 3, pp. 1446–1457, 2021.
- [24] Y. Tan, J. Qin, X. Xiang, C. Zhang, and Z. Wang, "Coverless steganography based on motion analysis of video," *Security and Communication Networks*, vol. 2021, Article ID 5554058, 16 pages, 2021.
- [25] N. Pan, J. Qin, Y. Tan, X. Xiang, and G. Hou, "A video coverless information hiding algorithm based on semantic segmentation," *EURASIP Journal on Image and Video Processing*, vol. 2020, no. 1, pp. 1–18, 2020.

- [26] M. Sandler, A. Howard, M. Zhu, and L.C. Chen, “Mobile-NetV2: inverted residuals and linear bottlenecks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, Seattle, WA, USA, June, 2018.
- [27] L. Zou, W. Wan, B. Wei, and J. Sun, “Coverless video steganography based on inter frame combination,” *Communications in Computer and Information Science*, vol. 1386, pp. 134–141, 2021.

Research Article

Detection of Face Morphing Attacks Based on Patch-Level Features and Lightweight Networks

Min Long ¹, Xuan Zhao,¹ Le-Bing Zhang,² and Fei Peng³

¹School of Computer and Communication Engineering, Changsha University of Science and Technology, Changsha 410114, China

²College of Computer Science and Engineering, Huaihua University, Huaihua 418000, China

³College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China

Correspondence should be addressed to Min Long; caslongm@aliyun.com

Received 5 November 2021; Accepted 3 February 2022; Published 11 March 2022

Academic Editor: Beijing Chen

Copyright © 2022 Min Long et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Aiming at the problem of face morphing attack detection under mobile and resource-constrained conditions, a face morphing detection method based on patch-level features and lightweight networks is proposed. It utilizes the combination of three blocks' structures for learning. By outputting the probability of each bona fide or morphed face patches, the whole face features are integrated for recognition. Experimental results and analysis show this method can significantly improve the detection accuracy of face morphing attacks. Compared with the existing methods, this method has the characteristics of high computational efficiency and strong robustness. It has great application potential in enhancing the security of the face recognition system.

1. Introduction

As an important biometric technology, face recognition has been widely used for banks, hotels, transportation, and other areas for identity verification. After the human's face was chosen by International Civil Aviation Organization (ICAO) as a biometric feature in electronic machine-readable travel documents (eMRTD) for assisting identity verification, face recognition technology was gradually applied to Automatic Border Control (ABC) system [1]. Recently, a variety of attacks against face recognition systems appeared, among which face morphing attack posed a serious threat to the security of the existing face recognition systems (FRS) [2].

Face morphs include splicing morphs, complete morphs [3], and combined morphs [4]. Generally, a morphed face image is generated by two subjects. Complete morphs are a result of warping and blending the entire image. Splicing morphs use the convex hull representing a face and it is cut from the input images. Combined morphs use Poisson image editing to hide face and background. Then warp and blend operations are performed. For splicing morphs and

combined morphs, the morphed image looks realistic because it is performed only in the face areas of two subjects, while for the complete morphs, the morphing operation is performed on the entire face, which usually leads to spurious shadows and tremendous visual inconsistencies in the hair region; therefore complete morphs are not appropriate for the morphing attack. Ferrara et al. showed the feasibility of face morphing attacks [2]. A criminal and an accomplice generate a morphed face image, and it is visually similar to the face images of criminal and accomplice, and it has both biological characteristics.

An example of a morphed facial image is shown in Figure 1. The pictures are from the publicly available face dataset Utrecht ECVP (<http://omen.cs.unimagdeburg.de/disclaimer/index.php>). If the morphed face image is used to apply for eMRTD, both the criminal and the accomplice can use this eMRTD to cross the boundary, as well as the FRS. Furthermore, the subsequent studies also proved the vulnerability of the FRS to face morphing attacks [5–7].

To countermeasure face morphing attacks, some methods have been proposed in recent years. Typical

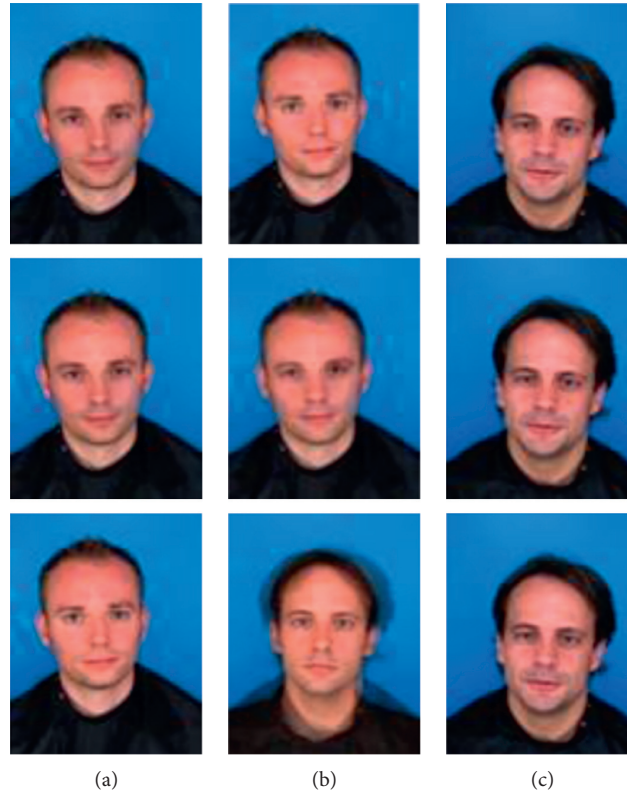


FIGURE 1: Example for a morphed face image (b) of subject 1(a) and subject 2(c). The middle face images in the first, second, and third row are called a splicing morphed image, combined morphed image, and complete morphed image, respectively. (a) Subject 1. (b) Morph. (c) Subject 2.

approaches include texture difference-based methods [8–17], image source feature-based methods [18–21], image quality-based methods [3, 22–24], and deep learning-based methods [25–37]. Among them, deep learning-based methods generally achieve good detection performance due to their strong feature description and learning ability, and deep learning can use nonlinear models to convert the original input data layer by layer into high-level abstract features. Therefore, methods based on deep learning have generally achieved good detection performance in the fields of image classification [38], image forensics [39, 40], face antispoofing [41, 42], and face morph attack detection. However, since deep learning models generally have large computing costs and high source requirements, the existing deep learning-based methods cannot be applied to resource-constrained application scenarios, such as mobile and embedded systems. To solve this problem, some lightweight networks such as SENet [43], SqueezeNet [44], MobileNetV2 [45], MobileNet-V3 [46], ShuffleNet [47], ShuffNetV2 [48], and Xception [49] have been successively proposed, and they are designed for common image recognition tasks.

To detect morphed face images, a novel method based on face patch-level features and lightweight networks is proposed in this paper. This method aims to make full use of the existing dataset and extract features by using lightweight convolutional network. The main contributions are summarized as follows.

- (i) A face patch-level feature learning approach is proposed and a two-level classification model is adopted. The use of patch-level features can expand the dataset and facilitate the extraction of identification information. The first-level classification uses the lightweight network to output the probability value of the face patch, and the second level of classification integrates the patch features of the whole face for discrimination. The two-level classification helps to improve the detection performance.
- (ii) A lightweight network architecture for morphing attack detection is designed. The network adopts a combination of three-block structures, and a lighter ECA-Net attention mechanism module is added to the inverted residual structure, which can reduce the number of network parameters and maintain detection accuracy.
- (iii) A new face morphed dataset named FERET_M is constructed based on the existing dataset. It includes 606 bona fide and 674 morphed face images. The experiments and cross-validation on three datasets demonstrate that the proposed method can achieve better face morphing attack detection performance compared with the existing methods.

The rest of the paper is organized as follows. The related work is introduced in Section 2. The proposed method is

described in Section 3. Experimental results and discussion are provided in Section 4. Finally, some conclusions and future works are drawn in Section 5.

2. Related Work

Currently, the existing face morphing attack detection methods can be divided into four categories according to the detection features used, namely, texture difference-based methods, image source feature-based methods, image quality-based methods, and deep learning-based methods.

2.1. Texture Difference-Based Method. In the proposed secure scan design, the test authorization code is used to manage scan operation. Normal scanning operation can only be enabled by entering the correct test authorization code. Due to the different texture features between the morphed face image and the bona fide face image, some image descriptors, such as Local Binary Patterns (LBP) [8], Binarized Statistical Image Features (BSIF) [9], and Histogram of Oriented Gradient (HOG) [10], have been successively used for face morphing attack detection. In [11], the authors proposed to identify the authenticity of the face image using BSIF and Support Vector Machine (SVM). Scherhag et al. proposed a multialgorithm fusion approach to detect morphing attacks using LBP, BSIF, HOG, and other features [12]. Then, a morphed face detection scheme based on hybrid color features was put forward in [13]. A novel algorithm is presented to detect the morphed images by leveraging the collaborative representation of microtexture features and deriving the information from different color spaces [14]. In [15], the authors designed a morph attack detection algorithm that leveraged an undecimated 2D Discrete Wavelet Transform (2D-DWT) for identifying morphed face images. Another work [16] also employed 2D-DWT to highlight inconsistencies between a real and a morphed image. In [17], a texture difference-based method was the morphed face detection using facial landmarks. The current texture difference-based methods were all tested by a single image dataset, and they have poor adaptability.

2.2. Image Source Feature-Based Method. Motivated by image source identification, Zhang et al. proposed to detect morphed faces by using the Fourier spectrum of sensor pattern noise (FS-SPN) [18]. FS-SPN is extracted based on guided image estimation, and the statistics of the specific frequency difference between the morphed face image and the bona fide face image are obtained and then input to SVM for morphed face detection. In [19–21], Photo Response Non-Uniformity (PRNU) is implemented for morphed face detection. More specifically, the method in [20] is mainly focused on the variance of PRNU-based features across image cells, while the method in [19] is mainly based on the analysis of spectral variation of PRNU caused by the morphing process. In [21], it analyzes the spatial and spectral features from PRNU patterns across image cells. Differences features between bona fide and morphed images are estimated during a threshold-selection stage using the Dresden

image database which is specifically built for PRNU analysis in digital image forensics. However, image source features-based methods are influenced by individual face morphing algorithms, and the detection accuracy needs to be improved.

2.3. Image Quality-Based Methods. Neubert et al. proposed to detect face morphing attacks based on image degradation [22]. By continuously analyzing the degradation process (e.g., JPEG compression) and manually generating several reference face images, the differences between the reference images and the original images are analyzed for face morphing attacks detection. Based on the observation that real face images comply with Benford's law, while morphed face images do not follow this law, Makrushin et al. proposed to detect face morphing forgeries based on Benford's law [23]. The detection is carried out by fitting a logarithmic curve to nine Benford features extracted from quantized DCT coefficients, which is the decomposition of JPEG compressed original and morphed face images. Besides, an automatic morphed face generation and detection of visually faultless facial morphs method was proposed in [3]. In [24], a demorphing approach is proposed. The live face image captured in real-time is subtracted from the morphed face image stored in the electronic document to obtain a demorphed image, and then the demorphed image is compared with the live face image by a specified threshold to determine whether it is the bona fide face. However, since the generation of the demorphed image depends on the prior knowledge of the morphing operation and parameters, the accuracy of demorphing is very limited.

2.4. Deep Learning-Based Methods. Presently, there are two types of approaches for detecting face morphing attacks using deep neural networks. One is to train a neural network from scratch, and the other is to use a pretrained neural network. In [25], a pretrained VGG19 [26] network was utilized to detect face morphing attacks. In [27], three convolutional neural networks, which were VGG19, GoogleLeNet [28], and AlexNet [29], were utilized for face morphing attacks detection. After that, another morphing attacks detection method based on CNNs was put forward in [30]. The feature level fusion of the first fully connected layers of VGG19 and AlexNet is specifically fine-tuned. In [31], the research showed that the facial features calculated by the general face recognition system can be used for morphing detection, while in [32], it was demonstrated that the features extracted by the deep learning-based FRS are feasible for differential morphing attack detection. A face morphing attack using Generative Adversarial Networks [33] was proposed in [34]. From a new perspective, an innovative demorphing approach using a Generative Adversarial Network, which was named as FD-GAN, was proposed by Peng et al. [35]. The symmetric dual network architecture and two-level recovery losses were utilized to separate the identity feature of the morphing accomplices. Besides, a partial face manipulation-based morphing attack was proposed to compromise the uniqueness of face

templates in [36]. The authors of [37] presented a novel differential morph detection framework by utilizing landmark and appearance disentanglement.

From the above analysis, the detection error rate of the general descriptors used for cross database face morphing attack detection is still high, and the generalization of the traditional algorithms is poor. The image source features-based methods are greatly affected by face morphing processes, and the detection performance of the image quality features-based methods is far from satisfactory. Although the deep learning-based face morphing detection methods can achieve good performance, it needs enough datasets support. Meanwhile, the datasets samples are limited in number of faces and attack types. If it is only effective for a single morphed attack method, the algorithm is easy to overfit. Furthermore, most of the existing methods utilize whole face as the input of face morphing detection, and the patch-level features of the face are not fully considered. Because some algorithms can only achieve good detection performance on a single dataset, it is necessary to improve the robustness and generalization ability across datasets. Therefore, a face morphing attack detection method based on patch-level features and lightweight networks is proposed in this paper to improve the generality of the algorithm.

3. The Proposed Method

In this section, we will introduce the details of the proposed method. The overall framework is illustrated in Figure 2, where face image preprocessing, lightweight network feature extraction, and two-level classification are the essential parts of the proposed method.

3.1. Face Image Preprocessing. The motivations of using patch-level features for image preprocessing are summarized as follows:

- (i) Discriminative information of face morphing detection is presented in the whole facial area, while the background region is redundant and will interfere with the information. The whole face is divided into several fixed nonoverlapping regions and the patch-level features of the face are used as the input, which is helpful to extract more identification information from the network. Furthermore, patch-level feature analysis has achieved good detection performance in image source feature-based methods [19, 20].
- (ii) The patch-level features can be used to perform data argumentation to datasets. For example, only 81 bona fide samples in FEI_M dataset are used for training. After a segmentation with a specific patch number N , the bona fide samples increase to $81 \times N$. Data augmentation can effectively increase the amount of training data and improve the generalization and robustness of the detection model based on deep learning.

- (iii) Face morphing detection can be regarded as a binary classification problem. The use of patch-level features can effectively train the network model proposed in this paper. Meanwhile, BagNet [38] shows that a powerful image classification model can be developed by using patch-level features.

Firstly, the face detection algorithm is used to detect the input face image. Figure 3 shows the processing procedure of the face datasets. For all bona fide or morphed faces, we use OpenCV tool library to detect faces and obtain face images and face rectangles. The face rectangle lacks a certain background area, so the interference information is reduced. Then, for the detected face images, the face location algorithm in the Dlib (<http://dlib.net/>) toolkit is used to detect 68 key points of the face, and the coordinates of the key points are obtained. After obtaining the face region according to the key point coordinates and face rectangle, the face is divided into different patches by using the face patch-level algorithm. Specifically, the length N of the local feature to be designed is set to be 16 in this paper. Then, according to the input requirements of the lightweight network, the cropped face area is determined by the coordinates of the eye key points, so as to obtain cropped local patch with the same size 96×96 pixels [41]. The usual patch-based approach divides the whole face into several fixed overlapping regions. Each patch is used to train an independent subnetwork. In this paper, for each image, we train a lightweight CNN on random patches extracted from the faces. We randomly crop the face region according to the position of the eyes without scaling the area and maintain the original resolution, so as to maintain the discrimination ability.

3.2. Lightweight Network. In the face morphing attacks detection algorithms, the networks with good detection performance are generally very complex and have high requirements for hardware resources. The existing face morph detection networks have the problems of large parameters and weak generalization ability. Therefore, this paper focuses on the lightweight of network. One way is to compress the trained network to obtain a relatively small model. Another way is to design a small, representative lightweight network design for training.

Therefore inspired by FeatherNets [42], a lightweight network is proposed. The following is a detailed introduction to the lightweight network.

3.2.1. The Components of the Proposed Network. The proposed lightweight network is shown in Figure 4. It consists of Block_1, Block_2, and Block_3. In particular, Block_1 is reverse residual structure mentioned in MobileNetV2 [45], which is mainly composed of Depthwise Separable Convolutions including a 1×1 pointwise convolution, 3×3 depthwise convolution, 1×1 pointwise convolution, and residual concatenation. Given the kernel size is $n \times n$, k and k' are the input and output channels, respectively. The parameter of standard convolution is $n \times n \times k \times k'$, and the Depthwise Separable Convolution is $n \times n + k \times k'$.

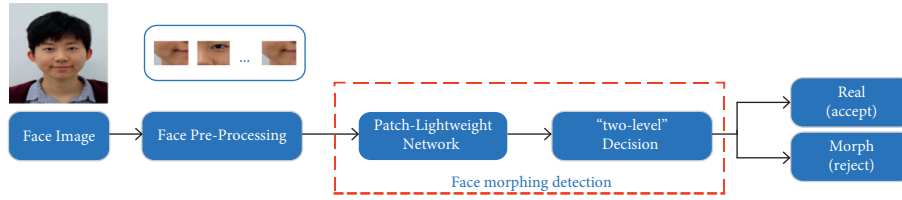


FIGURE 2: The overall framework of the proposed scheme.

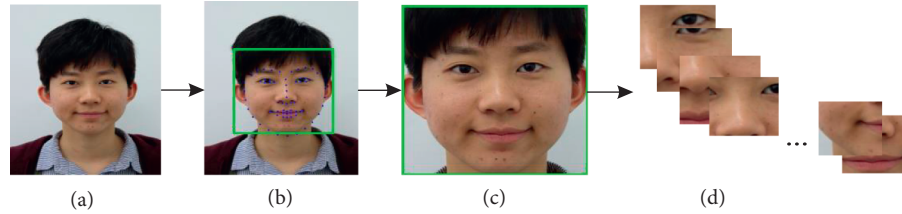


FIGURE 3: Preprocessing procedure of face dataset. (a) Input face image. (b) Face location and location of 68 key points. (c) Rectangular frame and face cropping. (d) Extract the face image patches; in this paper, the block length N is 16.

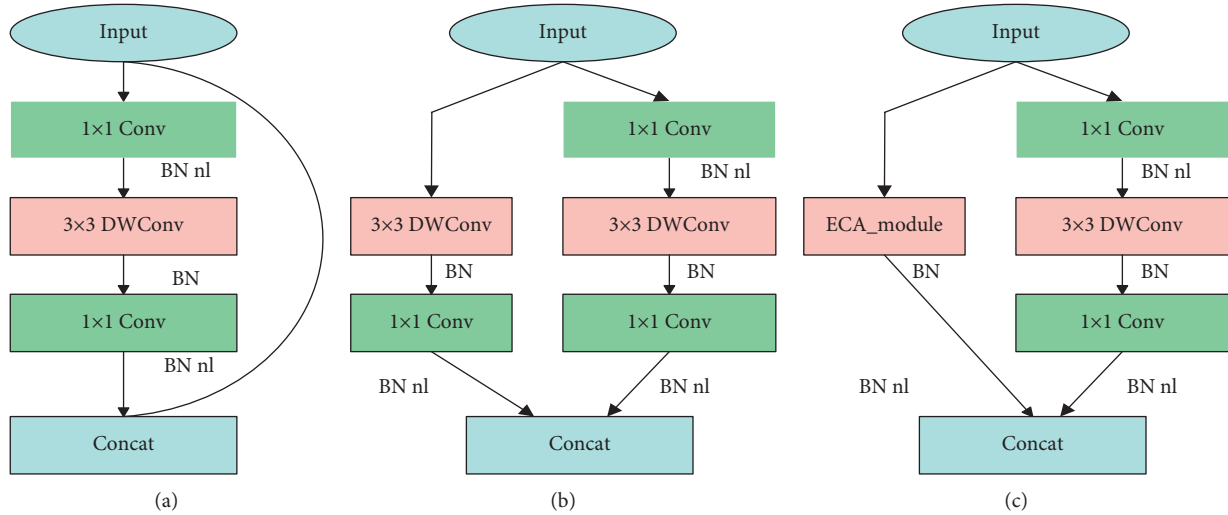


FIGURE 4: The main components of the proposed lightweight network. (a) Block_1. (b) Block_2. (c) Block_3.

Therefore, the deep separable convolution is used as the main convolution in this paper. Block_2 is composed of two parallel branches. The right branch is a Depthwise Separable Convolution, while the left branch includes a 3×3 depthwise convolution and a 1×1 pointwise convolution. The two branches are connected, and it can learn more features through the two branches. As illustrated in Xception [49], deepening the network width can improve the network performance. Thus, $\text{Stride}=2$ is assigned in the depthwise convolution of Block_2, and it is also named as the downsampling module. Here, Block_3 is specifically designed in this paper. In order to solve the problem of losing important feature information in the reduced convolution calculation, the channel attention module helps the information flow in the network by learning the information to be emphasized or suppressed. As a result, it adds the ECA-Net module [50] to the reverse residual structure.

Attention mechanisms are essentially similar to human selective visual attention. Generally speaking, it pays more

attention to key points and ignores other unimportant factors. For example, when we look at a picture of someone holding a table and a tree, when we look at it with our eyes, people will become the focus of what we see. But for machine detection, all objects are of equal importance.

As shown in Figure 3, although most face morph attack detection algorithms perform face cropping preprocessing steps, the importance of the four corners in the face and the image is different, because in the existing face fusion algorithms, only the face part is partially morphed, and the edge information such as hair is redundant information. Therefore, we believe that the attention mechanism can improve the detection effect of face morph attacks, allows the network to adaptively learn more important features, and is suitable for face morph attack detection, so we introduce the attention mechanism into the lightweight network.

ECA-Net is an improvement of SENet. It does not reduce the channel dimension, and it generates channel weights by performing 1D convolution with size k . Here, k is adaptively

determined by a function of channel dimension C , and it is a more lightweight channel attention mechanism, which has been successfully employed to improve the accuracy of CNN classification models. Therefore, the Block_3 can significantly reduce the number of parameters while maintaining performance gains. Attention-based layers can help to learn feature regions to detect bona fide and morphed faces. Thus, Block_1, Block_2, and Block_3 constitute the main components of the proposed lightweight network.

3.2.2. The Network Architecture. The details of the proposed lightweight network architecture are listed in Table 1. At the beginning of the network, the input dimension of the patch-level features of the face is $96 \times 96 \times 3$, and the regular convolution Conv2d (stride = 2) is used to extract and retain more features. The number of convolution kernels is 16, and the size of the convolution kernel is 3×3 . Then, the downsampling strategy of the Block_2 is used to reduce the input to $24 \times 24 \times 24$, which allows rapid reduction of the feature map size. It greatly reduces the calculation workload and parameters and can achieve faster speed. Block_1 module is the simplest one and it requires the least parameters among the three modules, so Block_1 is often used in the network. Block_3 is applied after Block_2 to minimize the loss of feature information due to downsampling of the channel attention mechanism. Block_3 can be used for all input channels above 40 to enhance the information interaction between deep network channels. Additionally, an ECA-Net module is also inserted after the second normal convolution layer to improve the accuracy of the model with appropriate parameters. The subsequent pool layer and the regular convolution layer are the final processes. The use of pool layers simplifies the model complexity and reduces computation and memory consumption. Finally, a fully connected layer acts as a classifier for the whole network.

In the deeper layers of the network, H -swish activation function [46] is used, and it is defined by

$$h\text{-swish}[x] = x \frac{\text{RELU6}(x+3)}{6}. \quad (1)$$

The use of H -swish can effectively improve the accuracy of deep networks. Therefore, due to the computation complexity, H -swish activation function is replaced with Rectified Linear Unit and Sigmoid only when the input channel of the proposed network is not less than 40. That is, in the proposed lightweight network, h -swish() is used when $nl = HS$, and the $\text{ReLU}()$ activation function is used when $nl = RE$.

3.3. Classification. The input faces will be divided into face patches with fixed size. For each patch, it will be input to the proposed lightweight network and converted into probability value of each face patch. The first-level judgment is performed on each patch to determine whether it is a bona fide one or a morphed one, and then the second-level judgment is performed to make a final decision. If the number of face patches determined as a bona fide one is greater than that of face patches

determined as morphed ones, this input face is determined as a bona fide face; otherwise, the input face is determined as a morphed face.

4. Experiment

In this section, we evaluate the effectiveness of the proposed method. Since complete morphs will lead to apparent artifacts, they are not suitable for a morphing attack unless they have undergone a manual retouch. Thus, the experiments are only carried out for splicing morphs and combined morphs.

4.1. Datasets. In the experiments, three face morph datasets FEI_M [18], HNU_FM [35], and a self-built FERET_M dataset are used.

Each of the three datasets is divided into a noncrossed training set, a validation set, and a test set, respectively. The training set is used to train the network model, the validation set is applied to adjust the hyperparameters, and the test set is used to verify performance of the model.

The FEI_M dataset is based on the public FEI face datasets, constructed according to the face morph algorithm in [3] with a fusion factor $\alpha = 0.5$. It consists of 200 subjects with 100 females and 100 males. FERET_M dataset is a new dataset built by us, and it is derived from the FERET dataset [51], which is a public face image dataset containing male and female images taken under different acquisition conditions with varying poses, facial expressions, and ages. The subjects suitable for face morph are manually selected from the color FERET dataset to build the new face morphing dataset with the standard morphing algorithm [4], where the pixel fusion factor α is fixed as 0.5. It consists of 303 subjects with 98 females and 205 males. The details of the FEI_M dataset and FERET_M dataset are listed in Table 2.

HNU_FM dataset is composed of four subdatasets FaceMDB1, FaceMDB2, FaceMDB3, and FaceMDB4, which are generated according to different pixel fusion factors α and location fusion factors β . There are 63 subjects, including 27 females and 36 males. A detailed description of the HNU_FM dataset is provided in Table 3. For FaceMDB1 dataset, the pixel fusion factors α and the location fusion factors β are both 0.5. For the FaceMDB2 dataset, the pixel fusion factor α is fixed to 0.5 and the location fusion factor β varies between 0.1 and 0.9. For the FaceMDB3 dataset, the pixel fusion factor α varies between 0.1 and 0.9, and the location fusion factor β is fixed to 0.5. For FaceMDB4, the pixel fusion factor α and the location fusion factor β both vary between 0.1 and 0.9.

4.2. Experimental Setup and Evaluation Criteria. Instead of the input of the traditional convolutional neural networks, the input size is 96×96 and Stochastic Gradient Descent (SGD) is utilized as the optimizer. The learning rate starts from 0.1 and it is adjusted using cosine annealing [52] with a cross-entropy loss function. Weight decay and momentum are set to 0.0001 and 0.9, respectively.

TABLE 1: The details of the proposed lightweight network.

Input	Operator	Exp size	#out	NL
$96 \times 96 \times 3$	Conv2d	—	16	RE
$48 \times 48 \times 16$	Block_2	24	24	RE
$24 \times 24 \times 24$	Block_1	24	24	RE
$24 \times 24 \times 24$	Block_3	24	24	RE
$24 \times 24 \times 24$	Block_2	96	40	RE
$12 \times 12 \times 40$	$4 \times$ Block_1	160	40	RE
$12 \times 12 \times 40$	Block_3	160	40	RE
$12 \times 12 \times 40$	Block_2	120	48	HS
$6 \times 6 \times 48$	Block_3	144	48	HS
$6 \times 6 \times 48$	Block_3	240	96	HS
$6 \times 6 \times 96$	Block_3	480	96	HS
$6 \times 6 \times 96$	Conv2d	—	256	HS
$6 \times 6 \times 256$	Pool, 6×6	—	—	—
$1 \times 1 \times 256$	Conv2d	—	1024	HS
$1 \times 1 \times 1024$	Conv2d	—	2	

TABLE 2: FEI_M dataset and FERET_M dataset.

Dataset	Subset	#Bona fide	#Splicing morphs
FEI_M	Training set	81	6480
	Validation set	20	380
	Testing set	99	3702
FERET_M	Training set	459	516
	Validation set	46	54
	Testing set	101	104

TABLE 3: HNU_FM dataset.

Dataset	Subset	#Bona fide	#Splicing morphs
FaceMDB1	Training set	1121	1121
	Validation set	564	330
	Testing set	566	377
FaceMDB2	Training set	1121	1125
	Validation set	564	567
	Testing set	566	567
FaceMDB3	Training set	1121	1125
	Validation set	564	567
	Testing set	566	567
FaceMDB4	Training set	1121	1134
	Validation set	564	567
	Testing set	566	567

The standardized ISO metrics Attack Presentation Classification Error Rate (APCER), Bona Fide Presentation Error Rate (BPCER), and Average Classification Error Rate (ACER) are used to evaluate the overall detection performance. APCER, BPCE, and ACER are, respectively, defined as follows:

APCER: proportion of attack presentations incorrectly classified as bona fide presentations in a specific scenario.

BPCER: proportion of bona fide presentations incorrectly classified as presentation attacks in a specific scenario.

ACER: the average of the sum of APCER and BPCER.

4.3. Experimental Results and Discussion. As complete morphs can lead to obvious artifacts unless they are manually trimmed, they are not suitable for morphing attacks.

Thus, the experiments are only performed to splicing morphs and combined morphs.

Similarly, the general steps of automatically generating the morphed facial image are as follows [35]. (a) Locate the key points of the morphed facial image, (b) warp the morphed facial image through Delaunay triangulation, and (c) blend the morphed facial image through pixel fusion.

For nondeep learning methods, public codes are often used for preprocessing and experiments [18]. According to the face deformation algorithm, the identification information of face deformation attack detection often exists in the face, and the background area is redundant, which is the interference information of face fusion attack detection. Therefore, in the preprocessing stage for the method based on deep learning, the face detector of Dlib library (<http://dlib.net/>) is used to detect the face of the input image,

and the detected face is aligned with the face. It combines with clipping operations to ensure that the algorithm is only used for face regions. In this paper we add patch-level processing to face preprocessing.

4.3.1. Single Dataset Test. To evaluate the detection performance of the proposed method, a comparative analysis is performed on the HNU_FM, FEI_M, and FERET_M datasets with 7 methods.

The experimental results on the HNU_FM dataset are listed in Table 4. It can be found that FS-SPN [18] can achieve the best APCER, while ResNet50 [53], MobileNetV2 [45], and the proposed method can obtain the best BPCER from FaceMDB1 to FaceMDB2 datasets (all are 0). MobileNetV2 [45] achieved the best ACER on FaceMDB2, FS-SPN [18] achieved the best ACER on FaceMDB4, and the best detection results on FaceMDB1 and FaceMDB3 belonged to the proposed method. Meanwhile, comparing the results on FaceMDB1 with those on FaceMDB3, the overall detection performance on FaceMDB3 is better than that on FaceMDB1. This is because the pixel fusion factor α of the FaceMDB3 dataset is not 0.5, and they are less likely to pass the face recognition system. In addition, the experimental results also demonstrate that the location fusion factor β has little impact on the detection.

The experimental results on FEI_M dataset are listed in Table 5. It can be found that FS-SPN [18] can achieve the best BPCER and ACER, while for the proposed method, it can obtain the best BPCER, and its ACER ranks the second after FS-SPN [18]. Since the number of bona fide on the FEI_M dataset is much smaller than the number of morphed face images, the samples of the dataset are unbalanced. Therefore, in general, BPCER is higher than APCER.

The experimental results on FERET_M dataset are listed in Table 6. It can be found the proposed method can achieve the best BPCER and ACER.

Considering the results on three image datasets, FS-SPN [18] can obtain the best performance among the nondeep learning methods. However, the detection performance of the deep learning-based methods is better than that of the nondeep learning methods on the whole. As the data volume is important to the performance of deep learning-based methods, the results on large sample image dataset are more convincing than those on small sample image dataset. The results of the proposed method on HNU_FM and FERET_M outperform most of the existing methods, which indicates the good performance of the proposed method on single dataset test.

4.3.2. Across Dataset Test. To evaluate the generality of the proposed method, comparative experiments are conducted to seven methods. In the across dataset test, one of three image datasets FEI_M, FaceMDB1, and FERET_M is used as training dataset, and the other two image datasets are used as test datasets; the results are listed in Tables 7–9, respectively.

As shown in Table 7, when the detection methods are trained by FEI_M and tested by FaceMDB1, it can be found that FS-SPN [18] can achieve the best APCER, while the

proposed method can obtain the best BPCER. When the detection methods are trained by FEI_M and tested by FERET_M, FS-SPN [18] can achieve the best APCER, while the proposed method can obtain the best BPCER and ACER.

Although the APCER of FS-SPN [18] is low, the BPCER is very high. The experimental results based on nondeep learning methods further illustrate that the generalization performance of traditional texture feature methods is poor. From the experimental results of ResNet [53] and MobileNetV2 [45], the method based on deep learning has good generalization ability.

Because of the imbalance of the FEI_M dataset, larger-scale samples may lead to overfitting and it is also easy to reduce the generalization ability of the model. When it is trained on the FEI_M dataset, the detection accuracy of most samples is high, and that of a few samples is low. Due to the use of a large number of datasets for enhancement, the method proposed in this paper suppresses this problem to a certain extent. Therefore, from the overall experimental results, a better ACER can be achieved.

As seen in Table 8, when the detection methods are trained by FaceMDB1 and tested by FEI_M dataset, it can be found that FS-SPN [18] can achieve the best APCER, MobileNetV2 [45], and the proposed method can obtain the best BPCER. MobileNetV2 [45] can achieve the best ACER. When the detection methods are trained by FaceMDB1 and tested by FERET_M, the proposed method can obtain the best BPCER and the best ACER.

From the overall experimental results in Table 8, the generalization performance of HOG [9], BSIF [10], and FS-SPN [18] tested on the FEI_M and FERET_M datasets is also very poor. The ACER results of the three methods are about 50%, which is close to random guess. The detection performance of the method based on deep learning is also better than the above three methods.

As seen in Table 9, when the detection methods are trained by FERET_M and tested by FEI_M, it can be found that BSIF [10] can achieve the best APCER, the proposed method can obtain the best BPCER, and ResNet50 [53] can achieve the best ACER. When the detection methods are trained by FERET_M and tested by FaceMDB1, BSIF [10] can achieve the best APCER and the proposed method can obtain the best ACER. MobileNetV2 [45] and the proposed method can obtain the best BPCER.

From the experimental results of Table 8 and Table 9, the difference between the datasets has a great impact on the detection results. When tested on the FaceMDB1 dataset (trained on the FERET_M dataset), the detection performance of all algorithms is much lower than that on the FERET_M dataset. This result is attributed to the difference between the bond fide of the two datasets.

In summary, when the detection methods are tested on FERET_M and FEI_M datasets, the overall detection performance is very poor due to the small number of the validation sets in the two datasets. Thus, it can be found that the number of positive and negative samples in the dataset has significant impact on the generalization performance of the face morphing attack detection, especially for methods based on deep learning. It can be also found that the model

TABLE 4: Performance comparison of different methods on HNU_FM dataset (%).

Method	FaceMDB1			FaceMDB2			FaceMDB3			FaceMDB4		
	APCER	BPCER	ACER	APCER	BPCER	ACER	APCER	BPCER	ACER	APCER	BPCER	ACER
HOG [9]	41.33	8.33	24.83	48.68	0.17	24.43	45.86	0.88	23.37	48.68	0.35	24.52
BSIF [10]	28.64	16.96	22.80	35.44	13.25	24.35	30.15	30.56	30.36	31.39	37.27	34.33
FS-SPN [18]	0	1.86	0.93	0	2.12	1.06	0	2.30	1.15	1.40	0.18	0.79
VGG19 [26]	27.32	1.22	14.27	23.28	0.88	12.08	37.21	0.35	18.78	36.86	1.06	18.96
ResNet50 [53]	3.98	0	1.95	2.12	0	1.06	2.47	0	1.24	3.35	0	1.68
SqueezeNet1_1 [39]	7.96	0	3.98	5.29	0.88	3.09	2.29	0.35	1.32	5.47	5.30	5.39
MobileNet V2 [40]	4.51	0	2.26	0.70	0	0.35	11.99	1.06	6.53	6.02	0	3.01
Proposed	1.33	0	0.67	1.76	0	0.88	1.23	0	0.62	1.59	0	0.80

TABLE 5: Performance comparison of different methods on FEI_M dataset (%).

Method	APCER	BPCER	ACER
HOG [9]	9.33	11.01	10.17
BSIF [10]	3.13	13.13	8.13
FS-SPN [18]	1.12	0	0.56
VGG19 [26]	1.94	4.04	2.99
ResNet50 [53]	0	3.03	1.56
SqueezeNet1_1 [44]	0.27	5.05	2.66
MobileNet V2 [45]	0	3.03	1.52
Proposed	1.98	0	0.99

TABLE 6: Performance comparison of different methods on FERET_M dataset (%).

Method	APCER	BPCER	ACER
HOG [9]	38.46	33.66	36.06
BSIF [10]	16.35	29.70	23.03
FS-SPN [18]	19.80	15.38	17.59
VGG19 [26]	23.07	17.68	20.38
ResNet50 [53]	12.73	7.07	11.80
SqueezeNet1_1 [44]	14.59	10.42	12.51
MobileNetV2 [45]	9.91	3.89	6.9
Proposed	9.91	1.98	5.95

TABLE 7: Comparative results of cross dataset test (trained by FEI_M) (%).

Method	Trained by FEI_M					
	Tested by FaceMDB1			Tested by FERET_M		
	APCER	BPCER	ACER	APCER	BPCER	ACER
HOG [9]	15.38	72.61	44.00	21.78	72.12	46.95
BSIF [10]	14.42	66.61	40.52	2.97	89.11	46.04
FS-SPN [18]	5.30	68.90	37.10	1.98	89.11	45.55
VGG19 [26]	18.04	57.07	37.55	25.96	69.31	47.64
ResNet50 [53]	7.96	42.40	25.18	8.65	50.50	29.58
SqueezeNet1_1 [44]	10.61	44.17	27.39	14.42	54.46	34.44
MobileNet V2 [45]	7.96	35.51	21.74	12.50	35.64	24.07
Proposed	21.22	0	10.61	27.72	0	13.86

TABLE 8: Comparative results of cross dataset test (trained by FaceMDB1) (%).

Method	Trained by FaceMDB1					
	Tested by FEI_M			Tested by FERET_M		
	APCER	BPCER	ACER	APCER	BPCER	ACER
HOG [9]	28.40	64.00	46.20	77.88	34.65	56.27
BSIF [10]	29.92	55.56	42.74	72.12	27.72	49.92
FS-SPN [18]	0.63	92.93	46.78	77.88	3.96	40.92
VGG19 [26]	25.77	45.45	35.61	52.88	39.60	46.24
ResNet50 [53]	15.46	30.30	22.88	37.50	25.74	31.62
SqueezeNet1_1 [44]	17.53	34.34	25.94	40.38	29.70	35.04
MobileNet V2 [45]	8.14	0	4.07	12.73	7.07	9.9
Proposed	9.07	0	4.54	8.65	3.89	6.27

TABLE 9: Comparative results of cross dataset test (trained by FERET_M) (%).

Method	Trained by FERET_M					
	Tested by FEI_M		Tested by FaceMDB1			
	APCER	BPCER	ACER	APCER	BPCER	ACER
HOG [9]	7.60	69.70	38.65	44.03	21.20	32.62
BSIF [10]	0.14	70.78	35.46	3.71	47.09	25.40
FS-SPN [18]	8.79	70.78	39.79	50.66	20.49	35.58
VGG19 [26]	19.57	50.50	34.81	26.79	27.56	27.18
ResNet50 [53]	9.27	35.35	22.31	14.85	16.43	15.64
SqueezeNet1_1 [44]	12.95	38.38	25.67	15.92	18.55	17.24
MobileNet V2 [45]	25.30	20.20	22.75	11.52	8.06	9.79
Proposed	73.23	2.02	37.63	7.96	8.06	8.01

TABLE 10: Comparison of Params and FLOPS.

Method	Params	FLOPS
VGG19 [26]	143.6M	19.67G
ResNet50 [53]	23.51M	3800M
SqueezeNet1_1 [44]	1.24M	357.30M
MobileNetV2 [45]	2.23M	306.17M
Proposed	0.57M	25.10M

strategy trained on the FERET_M dataset is better than the other two datasets. This is because the face images in FERET_M dataset contain more changes. The above results show that the generalization ability of deep learning-based methods outperforms nondeep learning methods, and the proposed method can achieve the better overall performance. However, it can be seen from the results that the proposed method is more biased towards bona fides, so the generalization performance still needs to be improved.

4.4. Analysis of Network Model Parameters. To further verify whether the proposed network meets the requirements of a lightweight network, the parameter sizes of different networks are compared and analyzed. The metrics for evaluating the size of network parameters include the parameter size of the network model generated by Pytorch (Parameter). In addition, the number of floating-point operations per second (FLOPS) is also presented. The Params and FLOPS for different models are listed in Table 10. It can be seen that the parameter size of the proposed method is only 0.57M and its FLOPS is only 25.10M, which are significantly lower than those of the other networks.

5. Conclusion and Future Works

In this paper, a face morphing attack detection method based on patch-level features and lightweight networks is proposed. By using patch-level features, not only is the dataset increased, but the ability of face representation is improved. Meanwhile, the proposed three-block combined lightweight networks help to reduce the number of network parameters. The experiments on 3 datasets and comparative analysis with some state-of-the-art methods show that the proposed method can achieve better detection performance with less network model parameters and operations. Furthermore, the cross datasets test also illustrates the good robustness ability of the proposed method. However, since

the dataset in this paper is not comprehensive enough and the postprocessing of morphing attack using digital images is not considered, the performance of the method needs further analysis and validation. In the evaluation criteria, the threshold relationship in the selected indicators is not considered. These are the places we will consider in the future. Our future work will also focus on how to enhance the robustness and generality of lightweight network against morphing attacks.

Data Availability

The data used to support the findings of the study are available from the corresponding author (caslongm@aliyun.com) upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (nos. 62072055, U1936115, and 92067104) and the Changsha Key Research and Development Program(no. kq2004004).

References

- [1] ICAO, *Biometric Deployment of Machine Readable Travel Documents*, TAG MRTDINTWG, Canada, 2004.
- [2] M. Ferrara, A. Franco, and D. Maltoni, "The magic passport," in *Proceedings of the IJCB*, pp. 1–7, Clearwater, FL, USA, 29 Sept.-2 Oct. 2014.
- [3] A. Makrushin, T. Neubert, and J. Dittmann, "Automatic generation and detection of visually faultless facial morphs," in *Proceedings of the 12th Int.Joint Conf. Comput. Vis. Imag. Comput. Graph. Theory Appl.*, pp. 39–50, 2017.
- [4] T. Neubert, A. Makrushin, M. Hildebrandt, C. Kraetzer, and J. Dittmann, "Extended StirTrace benchmarking of biometric and forensic qualities of morphed face images," *IET Biometrics*, vol. 7, no. 4, pp. 325–332, 2018.
- [5] D. J. Robertson, R. S. S. Kramer, and A. M. Burton, "Fraudulent id using face morphs: experiments on human and automatic recognition," *Plos One*, vol. 12, no. 3, p. e0173319, 2017.

- [6] M. Gomez-Barrero, C. Rathgeb, U. Scherhag, and C. Busch, "Is your biometric system robust to morphing attacks?" in *Proceedings of the IWBF*, pp. 1–6, Coventry, UK, 4-5 April 2017.
- [7] C. Rathgeb, K. Pöppelmann, and C. Busch, "Face morphing attacks: a threat to eLearning?" in *Proceedings of the EDUCON*, pp. 1149–1154, Vienna, Austria, 21-23 April 2021.
- [8] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," *Pattern Recognition*, vol. 29, no. 1, pp. 51–59, 1996.
- [9] C. Shu, X. Ding, and C. Fang, "Histogram of the oriented gradient for face recognition," *Tsinghua Science and Technology*, vol. 16, no. 2, pp. 216–224, 2011.
- [10] J. Kannala and E. Rahtu, "BSIF: Binarized statistical image features," in *Proceedings of the ICPR*, pp. 1363–1366, Tsukuba, Japan, 11-15 Nov. 2012.
- [11] R. Raghavendra, K. B. Raja, and C. Busch, "Detecting morphed face images," in *Proceedings of the BTAS*, pp. 1–7, Niagara Falls, NY, USA, 6-9 Sept. 2016.
- [12] U. Scherhag, C. Rathgeb, and C. Busch, "Morph detection from single face image: a multi-algorithm fusion approach," in *Proceedings of the ICBEA*, pp. 6–12, Amsterdam, Netherlands, May 16-18, 2018.
- [13] R. Ramachandra, S. Venkatesh, K. B. Raja, and C. Busch, "Towards making morphing attack detection robust using hybrid scale-space colour texture features," in *Proceedings of the ISBA*, pp. 1–8, Hyderabad, India, 22-24 Jan. 2019.
- [14] R. Ramachandra, K. B. Raja, S. Venkatesh, and C. Busch, "Face morphing versus face averaging: vulnerability and detection," in *Proceedings of the IEEE Int. Joint Conf. Biometr. (IJCB)*, pp. 555–563, Denver, CO, USA, 1-4 Oct. 2017.
- [15] B. Chaudhary, P. Aghdaie, S. Soleymani, J. Dawson, and N. M. Nasrabadi, "Differential morph face detection using discriminative wavelet sub-bands," in *Proceedings of the CVPRW*, pp. 1425–1434, Nashville, TN, USA, 19-25 June 2021.
- [16] P. Aghdaie, B. Chaudhary, S. Soleymani, J. Dawson, and N. M. Nasrabadi, "Detection of morphed face images using discriminative wavelet sub-bands," in *Proceedings of the IWBF*, pp. 1–6, Rome, Italy, 6-7 May 2021.
- [17] U. Scherhag, D. Budhrani, M. Gomez-Barrero, and C. Busch, "Detecting morphed face images using facial landmarks," in *Proceedings of the ICVIP*, Cherbourg, France, July 2018.
- [18] L. B. Zhang, F. Peng, and M. Long, "Face morphing detection using Fourier spectrum of sensor pattern noise," in *Proceedings of the ICME*, pp. 1–6, San Diego, CA, USA, 23-27 July 2018.
- [19] L. Debiase, U. Scherhag, C. Rathgeb, A. Uhl, and C. Busch, "PRNU based detection of morphed face images," in *Proceedings of the IWBF*, pp. 1–7, Sassari, Italy, 7-8 June 2018.
- [20] L. Debiase, C. Rathgeb, U. Scherhag, A. Uhl, and C. Busch, "PRNU variance analysis for morphed face image detection," in *Proceedings of the BTAS*, pp. 1–9, Redondo Beach, CA, USA, 22-25 Oct. 2018.
- [21] U. Scherhag, L. Debiase, C. Rathgeb, C. Busch, and A. Uhl, "Detection of face morphing attacks based on PRNU analysis," *IEEE Transactions on Biometrics, Behavior, and Identity Science*, vol. 1, no. 4, pp. 302–317, 2019.
- [22] T. Neubert, "Face morphing detection: an approach based on image degradation analysis," in *Proceedings of the IWDW*, pp. 93–106, Magdeburg, Germany, August 2017.
- [23] A. Makrushin, C. Kraetzer, T. Neubert, and J. Dittmann, "Generalized Benford's law for blind detection of morphed face images," in *Proceedings of the 6th ACM Workshop Inf. Hiding Multimedia Security (IH & MMSec)*, pp. 49–54, Innsbruck, Austria, June 20-22, 2018.
- [24] M. Ferrara, A. Franco, and D. Maltoni, "Face demorphing," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 4, pp. 1008–1017, 2018.
- [25] C. Seibold, W. Samek, A. Hilsmann, and P. Eisert, "Accurate and robust neural networks for security related applications exemplified by face morphing attacks," in *Proceedings of the CVPR*, pp. 1–16, Salt Lake City, UT, USA, June 2018.
- [26] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proceedings of the Comput. Vis. Pattern Recognit.*, ICLR, San Diego, CA, USA, May 2015.
- [27] C. Seibold, W. Samek, A. Hilsmann, and P. Eisert, "Detection of face morphing attacks by deep learning, digital forensics and watermarking," in *Proceedings of the IWDW*, pp. 107–120, Magdeburg, Germany, August 2017.
- [28] C. Szegedy, W. Liu, Y. Q. Jia et al., "Going deeper with convolutions," in *Proceedings of the CVPR*, pp. 1–9, Boston, MA, 7-12 June 2015.
- [29] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.
- [30] R. Raghavendra, K. B. Raja, S. Venkatesh, and C. Busch, "Transferable deep-CNN features for detecting digital and print-scanned morphed face images," in *Proceedings of the CVPRW*, pp. 1822–1830, Honolulu, HI, USA, 21-26 July 2017.
- [31] L. Wandzik, G. Kaeding, and R. V. Garcia, "Morphing detection using a general-purpose face recognition system," in *Proceedings of the EUSIPCO*, pp. 1012–1016, Rome, Italy, 3-7 Sept. 2018.
- [32] U. Scherhag, C. Rathgeb, J. Merkle, and C. Busch, "Deep face representations for differential morphing attack detection," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3625–3639, 2020.
- [33] I. Goodfellow, J. Pouget-Abadie, M. Mirza et al., "Generative adversarial nets," in *Proceedings of the NIPS*, pp. 2672–2680, Montreal, Quebec, Canada, December 2014.
- [34] N. Damer, A. M. Saladie, A. Braun, and A. Kuijper, "MORGAN: recognition vulnerability and attack detectability of face morphing attacks created by generative adversarial network," in *Proceedings of the BTAS*, pp. 1–10, Redondo Beach, CA, USA, 22-25 Oct. 2018.
- [35] F. Peng, L.-B. Zhang, and M. Long, "FD-GAN: face de-morphing generative adversarial network for restoring accomplice's facial image," *IEEE Access*, vol. 7, pp. 75122–75131, 2019.
- [36] L. Qin, F. Peng, S. Venkatesh, R. Ramachandra, M. Long, and C. Busch, "Low visual distortion and robust morphing attacks based on partial face image manipulation," *IEEE Transactions on Biometrics, Behavior, and Identity Science*, vol. 3, no. 1, pp. 72–88, 2021.
- [37] S. Soleymani, A. Dabouei, F. Taherkhani, J. Dawson, and N. M. Nasrabadi, "Mutual information maximization on disentangled representations for differential morph detection," in *Proceedings of the WACV*, pp. 1730–1740, Waikoloa, HI, USA, 3-8 Jan. 2021.
- [38] W. Brendel and M. Bethge, "Approximating CNNs with Bag-of-Local-Features models works surprisingly well on ImageNet," in *Proceedings of the ICLR*, New Orleans, LA, USA, May 2019.
- [39] B. Chen, W. Tan, G. Coatrieux, Y. Zheng, and Y.-Q. Shi, "A serial image copy-move forgery localization scheme with source/target distinguishment," *IEEE Transactions on Multimedia*, vol. 23, pp. 3506–3517, 2021.

- [40] B. Chen, X. Liu, Y. Zheng, G. Zhao, and Y.-Q. Shi, "A robust GAN-generated face detection method based on dual-color spaces and an improved Xception," *IEEE Transactions on Circuits and Systems for Video Technology*, p. 1, 2021.
- [41] Y. Atoum, Y. Liu, A. Jourabloo, and X. Liu, "Face anti-spoofing using patch and depth-based CNNs," in *Proceedings of the IJCB*, pp. 319–328, Denver, CO, USA, 1-4 Oct. 2017.
- [42] P. Zhang, F. Zou, Z. Wu et al., "FeatherNets: convolutional neural networks as light as feather for face anti-spoofing," in *Proceedings of the CVPRW*, pp. 1574–1583, Long Beach, CA, USA, 16-17 June 2019.
- [43] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, "Squeeze-and-excitation networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 8, pp. 2011–2023, 2020.
- [44] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: alexnet-level accuracy with 50x fewer parameters and 1mb model size," in *Proceedings of the ICLR*, Toulon, France, April 2016.
- [45] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "MobileNetV2: inverted residuals and linear bottlenecks," in *Proceedings of the CVPR*, pp. 4510–4520, Salt Lake City, UT, USA, 18-23 June 2018.
- [46] A. G. Howard, M. Sandler, G. Chu et al., "Searching for MobileNetV3," in *Proceedings of the ICCV*, pp. 1314–1324, Seoul, Korea, 27 Oct.-2 Nov.
- [47] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shuffle-net: an extremely efficient convolutional neural network for mobile devices," in *Proceedings of the CVPR*, Salt Lake City, USA, June 2018.
- [48] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "Shufflenet v2: practical guidelines for efficient cnn architecture design," in *Proceedings of the ECCV*, pp. 122–138, Munich, Germany, September 2018.
- [49] F. Chollet, "Xception: deep learning with depthwise separable convolutions," in *Proceedings of the CVPR*, pp. 1800–1807, Honolulu, HI, USA, 21-26 July 2017.
- [50] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, "ECA-net: efficient channel attention for deep convolutional neural networks," in *Proceedings of the CVPR*, pp. 11531–11539, Seattle, WA, USA, 13-19 June 2020.
- [51] P. J. Phillips, H. Wechsler, J. Huang, and P. J. Rauss, "The FERET database and evaluation procedure for face-recognition algorithms," *Image and Vision Computing*, vol. 16, no. 5, pp. 295–306, 1998.
- [52] G. Huang, Y. Li, G. Pleiss, Z. Liu, J. E. Hopcroft, and K. Q. Weinberger, "Snapshot ensembles: train 1, get M for free," in *Proceedings of the ICLR*, Toulon, France, April 2017.
- [53] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the CVPR*, pp. 770–778, Las Vegas, NV, USA, June 2016.

Research Article

Toward Steganographic Payload Location via Neighboring Weight Algorithm

Tong Qiao ^{1,2}, Xiangyang Luo ², Binmin Pan,¹ Yuxing Chen,¹ and Xiaoshuai Wu¹

¹School of Cyberspace, Hangzhou Dianzi University, Hangzhou, China

²State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou Science and Technology Institute, Zhengzhou, Henan, China

Correspondence should be addressed to Xiangyang Luo; xiangyangluo@126.com

Received 12 December 2021; Accepted 31 January 2022; Published 25 February 2022

Academic Editor: Beijing Chen

Copyright © 2022 Tong Qiao et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Modern steganalysis has been widely investigated, most of which mainly focus on dealing with the problem of detecting whether an inquiry image contains hidden information. However, few articles in the literature study the location of secret bits hidden by modern adaptive steganography. In this paper, we propose a novel algorithm for locating steganographic payload in the spatial domain. We first predict the steganographic scheme and its payload, which is used for generating a random bitstream. Then, the random bits are embedded in the stego image based on the cost matrix in the framework of Syndrome-Trellis Codes (STCs). Next, relying on the differences between two stego images, the extended modification map in couple with the neighboring weight algorithm can be acquired, leading to the location of the hidden bits. Compared with the prior art, the extensive experiments verify that our proposed locating algorithm performs better, in terms of locating accuracy and efficiency.

1. Introduction

Steganography is the science and art of covertly transmitting the secret message in a carrier, such as widely adopted multimedia content. In general, an empirical cover carries the secret message under the supervision of the warden while the recipient extracts it to accomplish covert communication. In the past two decades, image steganography has made great progress. To counter against steganography, the analysis technique of detecting a steganographic image, defined as steganalysis, has also been advanced.

To ensure the undetectability of image steganography, a practical and common manner is to change cover pixels slightly by ± 1 . In particular, in the early stage of the study in this field, LSBR (Least Significant Bit Replacement) is designed, which randomly spreads the modification changes to the whole cover image; LSBM (Least Significant Bit Matching) is proposed to avoid the asymmetry artifacts by randomly modifying LSBs. In the current image steganography, the study of image content-adaptive schemes is usually given the first priority. One of the most successful

adaptive models rather treats the message embedding as a source coding problem with a fidelity constraint [1], instead of taking the cover source distribution into account. In this framework of minimizing the distortion caused by embedding, the establishment of the cost function becomes fundamentally important for the steganographer who prefers hiding information in the texture region of a cover image.

Many modern adaptive steganographic algorithms have been proposed, such as in spatial domain Highly Undetectable steGo (HUGO) [2], Wavelet Obtained Weights (WOW) [3], Spatial UNiversal WAVElet Relative Distortion (S-UNIWARD) [1], HIgh-pass, Low-pass, and Low-pass (HILL) [4], and in JPEG domain JPEG Uiversal WAVElet Relative Distortion (J-UNIWARD) [1], Uniform Embedding Distortion (UED) [5], and Uniform Embedding Revisited Distortion (UERD) [6]. Moving the study from laboratory to real world, however, most of the current steganographic methods have poor performance of resisting JPEG compression or rescaling attack. Thus, some robust steganalysis detectors are recently proposed to address that challenge

such as [7–10]. The steganographic algorithms always aim to hide the secret information in an imperceptible manner to ensure that the stego image visually and statistically behaves very similar to its counterpart cover source.

In the face of the challenge proposed by steganography, the task of steganalysis is to classify between the cover and stego source. Specifically, in the generalized framework of steganalysis, steganalysis aims to (1) detect the existence of hidden information (see [11–14]), namely, binary classification, (2) predict the size of the payload, also defined as quantitative steganalysis (see [15]), (3) locate the steganographic payload, and (4) extract the secret information (see [16–18]), also defined as forensic steganalysis. In the recent studies of steganalysis, most researchers focus on detecting if the secret information is hidden in an image. Relying on the rich models, together with the ensemble learning-based mechanism, the state of the arts (see [19–22]) perform very well in dealing with the problem of classifying between cover and stego images. Recently, in the framework of deep learning [23, 24], instead of hand-crafted feature extraction, the realization of end-to-end automatic image steganalysis gradually becomes widespread (see [25–33]). Furthermore, quantitative steganalysis algorithms have also been investigated (see [34]).

In this paper, we mainly study the algorithm of payload location, which currently receives less attention compared with both binary classification and quantitative steganalysis. By predicting the cover source (or specifically by calculating the differences between inquiry stego and predicted cover source), a series of steganalysis locators targeting LSBR or/and LSBM embedding steganography have been designed, such as [35–39]. Without loss of generality, the problem of locating hidden bits can be smoothly transferred as binary classification. Based on the prescribed threshold, each pixel is classified as an innocent or stego sample. Also, the following established algorithms obey the rule of binary classification by using hand-crafted SPAM features [40] of [41] or deep-learning-based features [42]. Recently, [43, 44] propose locating hidden bits in the DCT domain, mainly targeting JSteg and F5 steganography, whose stego key can be recovered in [45].

In fact, most prior locating algorithms have two remarkable limitations. When dealing with modern adaptive steganography, it probably becomes invalid. Furthermore, most algorithms are designed for one targeted steganography, such as LSBR or LSBM, which cannot be used for universal location. To overcome the current limitations, let us establish a universal detector of locating the payload of adaptive steganography, only dependent on a single inquiry image. The core idea behind our proposed algorithm is that modern adaptive steganography is prone to embed secret bits into the texture region of an image. It should be noted that our proposed algorithm can only locate the flipped hidden bits (± 1 happens) not including nonflipped bits. In fact, when the embedding procedure cannot modify the bits of a cover image, those nonflipped bits are hard to be located due to their unchanged property during embedding. Then, the main contributions are listed in the following:

- (1) In virtue of the intrinsic property of adaptive steganography, we propose to design the steganalysis algorithm toward payload location relying on a single inquiry image
- (2) Based on the proposed neighboring weight algorithm (NWA), we establish the extended modification map and its refined version for predicting the flipped-hidden-bit location, which further narrows down the prediction error and improves the location accuracy
- (3) For practical use, we propose four cases of locating steganographic payload, referring to as KPKS, UPKS, KPUS, and UPUS (see details in Table 1)
- (4) Numerical experiments empirically verify the effectiveness of the proposed location algorithm, which can deal with different modern adaptive steganographic algorithms such as WOW, S-UNIWARD, and HILL. Moreover, compared with the prior arts, our scheme performs its superiority

The rest of the paper is organized as follows. We first overview the state of the arts concerning the study of locating hidden bits. We present the core idea of designing a detector for payload location in Section 3. In Section 4, the detailed steps of locating hidden bits by adaptive steganography are extended. Furthermore, our proposed neighboring weight algorithm is specifically described. Next, the numerical experimental results are provided in Section 5, including the evaluation of our proposed algorithm as well as the comparison with the prior arts. Finally, we conclude this paper in Section 6.

2. State of the Arts

In this paper, we mainly focus on the study of locating payload. In general, the problem of locating hidden bits is always solved by biclassifying each pixel of the inquiry image. For clarity, let us define a cover image as a vector $\mathbf{c} = \{c_l\}$, $l \in \{1, \dots, L\}$ and a corresponding stego image described as a vector $\mathbf{s} = \{s_l\}$, $l \in \{1, \dots, L\}$. Then, the discrimination factor d_l , denoted as residual noise between stego and predicted cover source, is formulated as

$$d_l =_{\text{idc}} f l \in \{1, \dots, L\}$$

$$\text{with } f_{\text{idc}}[a, b] = \begin{cases} 1, & \text{if } a \neq b, \\ 0, & \text{otherwise,} \end{cases} \quad (1)$$

where the indicator function $f_{\text{idc}}[\cdot]$ is used to label the predicted pixel with/without hidden bits, and the estimated cover pixel is denoted as \hat{c}_l . On the assumption that all the hidden bits are embedded in the same position for a number of images, it holds true that the stego pixels can be successfully located by averaging the d_l . In this scenario, the expected value of the averaged differences is denoted as $E[\overline{d_l}]$. In detail, when the cover pixel is used for embedding, including the cases of flipping and nonflipping, $E[\overline{d_l}]$ equals 0.5; when the cover pixel is not selected for embedding, $E[\overline{d_l}]$ equals 0. Next, by calculating the average value of each pixel

TABLE 1: Abbreviation of location scenarios in the framework of our proposed algorithm.

	Known scheme	Unknown scheme
Known payload	KPKS	KPUS
Unknown payload	UPKS	UPUS

among a group of stego images, we can predict the position of hidden bits. Meanwhile, the optimal threshold arrives at 0.25 lying between two expected values. Immediately, let us extend the related works of the locating algorithms.

In [35], inspired by the Weighted Stego (WS) image steganalysis method, a steganalysis algorithm is designed for locating hidden bits embedded by LSBR. Specifically, the linear filter is used for predicting the average residual value of each pixel, in which a residual-based threshold is empirically prescribed for locating embedding positions. Next, to deal with the problem of LSBM steganography, [36] proposes adopting a Wavelet Absolute Moment (WAM) filter to extract the residual, which characterizes the distinguishable features between the cover and stego pixels. By estimating \hat{c}_i or directly calculating the \bar{d}_i , two aforementioned methods have verified their effectiveness in locating hidden bits. Although the payload of the old steganography is successfully located, the limitations of the aforementioned methods are as follows: a large number of stego images have to share the same size; the secret bits are hidden in the same positions for each image; a locating algorithm is only applicable for one targeted steganographic algorithm, such as LSBR or LSBM.

In the following studies, high-order features represented by residuals are used to locate the steganographic payload embedded by LSBM (see [37]). To unify the location of hidden bits embedded by LSBR or LSBM, relying on the theory of Maximum A Posteriori (MAP) [38] designs a detector which remarkably improves the location performance. The accurately estimated cover source brings more discriminative residuals, which straightforwardly leads to improved location accuracy. Next, [39] designs an effective algorithm to extract the hidden bits independent of the embedding key. In fact, the algorithms mainly put the focus either on the estimation of cover source \hat{c}_i [38, 39] or directly on calculating the weighted average of residual noise [37]. It makes sense that the fidelity of the cover source estimation directly impacts the accuracy of payload location (see [38] for details). Nevertheless, the methods still need large-scale stego images within embedding the same payload location.

Inspired by the work [40], [41] proposes dealing with the location problem by classifying each pixel into binary types: payload and nonpayload. Although the framework equation (1) is not used, the hidden bits can be successfully located by investigating the features of each pixel. The discriminative features (72-dimensional features for each pixel) characterized by neighboring pixel-value differences are used for training a Support Vector Machine (SVM) classifier, which serves for binary classification during the stage of locating hidden bits. Although the learning-based method improves the accuracy and efficiency of locating, the performance is degraded when the payload is increased. Recently, to solve

the problem of inaccuracy location within the small payload, [42] proposes an efficient detector for locating hidden bits relying on the deep neural networks. However, still, it can only be applied to the stego image generated by old steganography, such as nonadaptive LSBM.

To our knowledge, few studies focus on locating adaptive steganographic payload. The article [46] opens a way to investigate the location of the steganographic payload embedded by modern adaptive algorithms. By reembedding randomly generated bits into the stego image, the hidden bits are generally located. However, when both the embedding scheme and the size of the payload are unknown, the accuracy of location cannot be guaranteed. Thus, in this paper, to further improve the location accuracy and reduce the prediction error, let us design an effective detector based on the proposed neighboring weight algorithm (NWA).

3. Statement of the Problem

In the community of data hiding, most literature studies focus on the establishment of locating algorithms for nonadaptive steganography while the challenging problem of payload location for adaptive steganography has not been widely investigated. For simplicity and clarity, it is proposed to illustrate the pipeline of our locating algorithm (see Figure 1). When an inquiry image is used for location, we first have to estimate its embedding payload and predict the embedding scheme. Because our proposed algorithm only works well in the scenario that the inquiry image has been confirmed as stego one with acquiring its steganographic scheme, subsequently, we can generate a random bitstream based on the payload. Then, let us reembed the random bits into the stego image based on the cost matrix in the framework of STCs. Next, relying on the differences between two stego images, the modification map can be obtained. By using the proposed neighboring weight algorithm, the modification map is further extended. Finally, our proposed algorithm is capable of locating flipped bits. For clarity, the main mathematical notations used in this paper are summarized in Table 2.

3.1. Establishment of Modification MAP. By modifying pixels within texture regions, modern adaptive steganography performs very well and especially remains its high undetectability. That property inspires us to investigate if the modified pixels are selected again when reembedding happens, meaning that the locations of the steganographic payload are overlapped between a stego image and its reembedding version. That is because the cost matrix of the two images nearly remains unchanged.

First, let us embed a random bitstream into a grey-level cover image $\mathbf{C} = \{c_{i,j}\}$, $i \in \{1, \dots, I\}$, $j \in \{1, \dots, J\}$, leading to the generation of a stego image $\mathbf{S}^{(1)} = \{s_{i,j}^{(1)}\}$. Next, by modifying the original stego image $\mathbf{S}^{(1)}$ acquired from \mathbf{C} , let us use the same bitstream to generate a new stego image $\mathbf{S}^{(2)} = \{s_{i,j}^{(2)}\}$. It should be noted that regardless of hidden bits (the same, flipped or random ones), we denote the first stego image generated from the cover \mathbf{C} as $\mathbf{S}^{(1)}$ while the second stego image from $\mathbf{S}^{(1)}$ as $\mathbf{S}^{(2)}$. It is worth noting that

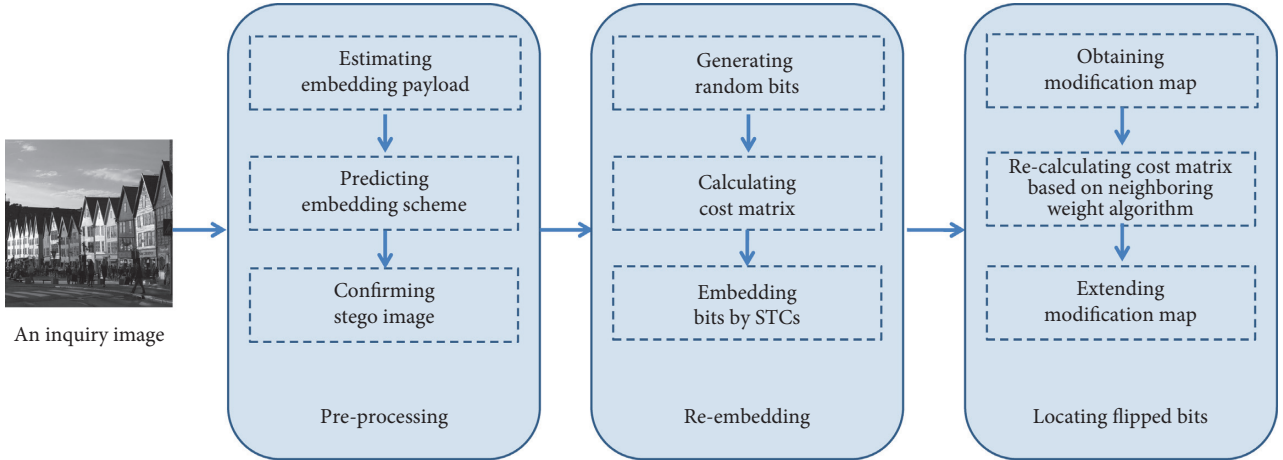


FIGURE 1: Pipeline of our proposed method.

TABLE 2: Notations.

C	Grey-level cover image
$S^{(1)}$	Original stego image by the first embedding
$S^{(2)}$	New stego image by reembedding
$M^{(1)}$	Modification map
$M^{(e)}$	Extended modification map
$M^{(a)}$	Refined modification map
m	Hidden bits
ρ	Cost matrix
ω	Weight factor
d	Euclidean distance

in the framework of STCs, the cost function (see [1] for instance) guides us to select the ready-to-embed pixels and to generate both $S^{(1)}$ and $S^{(2)}$. Then, the modification map can be straightforwardly formulated as

$$M^{(1)}(i, j) = \begin{cases} 255, & \text{if the pixel } c_{i,j} \text{ is flipped,} \\ 0, & \text{otherwise,} \end{cases} \quad (2)$$

where both cover and stego images share the same size of the modification map $M^{(1)}$. As Figure 2 illustrates, an 8-bit cover image with the size of 512×512 and the modification maps come from its corresponding stego images. $M^{(1)}$ is acquired by making difference between C and $S^{(1)}$, which visually labels the flipped pixels caused by embedding. Meanwhile, $M^{(2)}$ can be obtained from both $S^{(1)}$ and $S^{(2)}$. It should be noted that four different practical scenarios are considered in our proposed algorithms, referring to KPUS, UPUS, KPUS, and UPUS (see details in Table 1 of Section 3.2). Two embedding operations both prefer embedding the bits nearly at the same locations, referring to the texture region. Furthermore, few locations have the value 255 in the same position of two modification maps, meaning that few pixels at the positions experience twice modification. Thus, we need to extend the modification map $M^{(2)}$ for digging out more hidden bits (see details in Section 4).

3.2. Practical Scenario of Locating Steganographic Payload. In this paper, to overall evaluate the effectiveness of the proposed locating algorithm, we intend to address that

challenging problem in the four practical scenarios (see Table 1 for details). Before locating the flipped bits by adaptive steganography, it is proposed to emphasize a prerequisite that the inquiry image has been detected as stego one with secret information. Thus, we list two assumptions: *known payload* or *unknown payload*. When the payload is known, the steganalyst is capable of conducting the locating algorithm straightforward; when the payload is unknown, the prediction algorithm (or defined as quantitative steganalysis), such as [15], has to be conducted first. It is worth noticing that when the predicted payload α is larger than the given threshold τ , the inquiry image is detected as stego. Besides, in the procedure of locating flipped bits using the proposed algorithm, reembedding is obligatory. However, when a stego image is obtained, it hardly holds true that we can acquire the embedding scheme used for the stego image. Thus, another two assumptions should also be addressed, referring to as a *known scheme* or an *unknown scheme*. The specific experimental results are extended in Section 5.

In the framework of our proposed locating algorithm, the key point is how to confirm the adjacent regions for reembedding. If the large size of the adjacent region is selected, many incorrectly classified pixels will be included, leading to the decreased accuracy of the location. On the contrary, if the small size of the adjacent region is selected, possibly some missing-classified pixels that are actually flipped by adaptive steganography cannot be accurately located. To deal with that trade-off problem, we thus propose improving the performance of locating hidden bits based on the neighboring weight algorithm. In the following section, we first specifically describe our proposed locating algorithm. More importantly, the NWA is designed to further reduce location errors.

4. Proposed Work

In this section, we first introduce the general steps of locating a steganographic payload algorithm. Next, the establishment of the extended modification map is specifically presented.

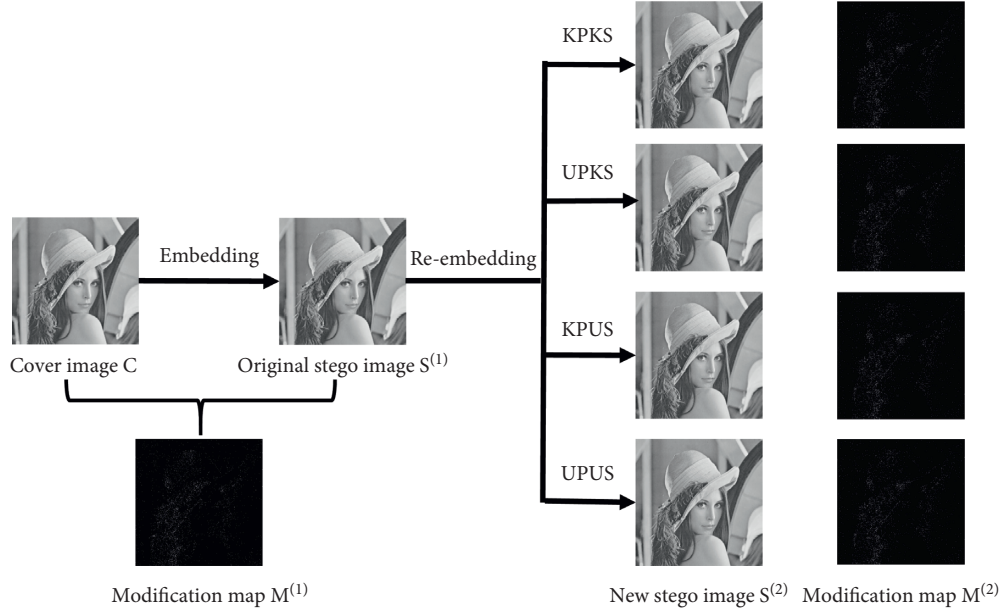


FIGURE 2: Illustration of our proposed framework.

Then, we develop two novel schemes dealing with the problem of refining the extended modification map.

4.1. Description of Our Locating Algorithm. The description of our locating algorithm can be summarized in Algorithm 1.

Generating a random bit stream: a random bitstream \mathbf{m} with the length L is generated. Note that $L = \text{Num} \times \alpha$ where Num denotes the total amount of pixels, and α is the relative payload. **Calculating the cost matrix:** relying on an adaptive steganographic algorithm, a bank of designed filters is then utilized to obtain the cost matrix of the stego image. For simplicity and clarity, let us denote the stego image as \mathbf{S} , and the cost matrix as ρ referring to [1]. **Embedding secret message using STCs:** without loss of generality, STCs are used to embed message \mathbf{m} into the stego $\mathbf{S}^{(1)}$ on the principle of minimizing the distortion function based on the cost matrix ρ . In such a manner, the stego version of image $\mathbf{S}^{(1)}$ after modification is denoted as $\mathbf{S}^{(2)}$. **Obtaining the modification map:** based on the differences between the two stego images, the modification map \mathbf{M} is obtained as described in Section 3.1. **Extending the modification map:** to locate hidden bits, we intend to extend the modification map $\mathbf{M}^{(o)}$ to $\mathbf{M}^{(e)}$ in a given margin value N . **Refining the extended modification map:** to locate the modified pixels as more as possible and reduce the number of incorrectly predicted bits, the refined extended modification map $\mathbf{M}^{(o)}$ is established based on the redesigned cost matrix ρ' .

Without loss of generality, the selection of N , denoted as margin value, is actually a trade-off problem in the design of the extended modification map. In detail, the value N would increase if we intend to locate the modified pixels as more as possible. While as the margin value N becomes larger, more and more innocent pixels (without being modified) would be also involved. In this context, the designed extended

modification map should follow two requirements: (1) more hidden bits are contained in the map, denoting the location; (2) less innocent bits are excluded in the map.

In the following sections, let us specifically introduce the design of the extended modification map. More importantly, based on the proposed neighboring weight algorithm, the map is further refined for locating more hidden bits and abandoning more innocent bits.

4.2. Design of Extended Modification MAP. In fact, based on the intrinsic property of the content-adaptive scheme, it hardly holds true that modern adaptive steganography modifies the pixel of the same location twice when embedding the same random bits. And meanwhile, the adjacent region of the pixel modified by the first embedding probably contains the modified pixels caused by the second embedding. Immediately, based on the results of Figure 2, it is proposed to extend the $\mathbf{M}^{(1)}$ by covering each pixel's neighbors, that is formulated as

$$\mathbf{M}^{(e)}(i+p, j+p) = \begin{cases} 255, & \text{if the pixel at } (i, j) \text{ is flipped,} \\ 0, & \text{otherwise,} \end{cases} \quad (3)$$

where $p \in [-N, N]$ represents an integer controlled by the extension maximum, margin value N . Next, the adjacent regions of a pixel in variant margin value N are illustrated in Figure 3(a). Figure 3(b) illustrates $\mathbf{M}^{(e)}$ with the margin value $N = 3$, where the bright regions (the pixels in the regions equal to 255) definitely cover a large portion of pixels flipped by the first embedding. When the margin value equals N , the size of its adjacent region is calculated by the following function $(2N + 1) \times (2N + 1)$. Obviously, when $N = 0$, the modification map $\mathbf{M}^{(e)}$ is equivalent to $\mathbf{M}^{(2)}$. Furthermore, let us define the rate $r = m/n$, where n is the

Input: Stego image $\mathbf{S}^{(1)}$, steganographic payload α

Output: Predicted locations of steganographic payload

- (1) //Generating a random bit stream $\mathbf{mNum} = f_{\text{num}}[\mathbf{S}^{(1)}]$, function $f_{\text{num}}[\cdot]$ for calculating the number of input data $L = \text{Num} \times \alpha$, denoting the number of bits; $\mathbf{m} = G[L]$, function $G[\cdot]$ for generating random bits
- (2) //Calculating the cost matrix ρ
- (3) //Embedding secret message $\mathbf{mS}^{(2)} = f_{\text{emb}}[\mathbf{S}^{(2)}, \mathbf{m}, \rho]$, function f_{emb} is used for embedding bits \mathbf{m} into $\mathbf{S}^{(1)}$ in the framework of STCs
- (4) //Obtaining the modification map $\mathbf{M}^{(1)} \leftarrow f_{\text{diff}}[\mathbf{S}^{(1)}, \mathbf{S}^{(2)}]$, function $f_{\text{diff}}[\cdot]$ calculates the differences between $\mathbf{S}^{(1)}$ and $\mathbf{S}^{(2)}$ to generate the modification map
- (5) //Extending the modification map $\mathbf{M}^{(e)} \leftarrow \mathbf{M}^{(1)}$
- (6) //Refining the extended modification map $\mathbf{M}^{(o)} = f[\mathbf{M}^{(e)}]$ by redesigning the cost matrix ρ' , function $f[\cdot]$ refines the original extended modification map $\mathbf{M}^{(e)}$. $\mathbf{M}^{(o)}$ labels all the predicted locations via our proposed algorithm

ALGORITHM 1: Procedure to Locate Steganographic Payload.

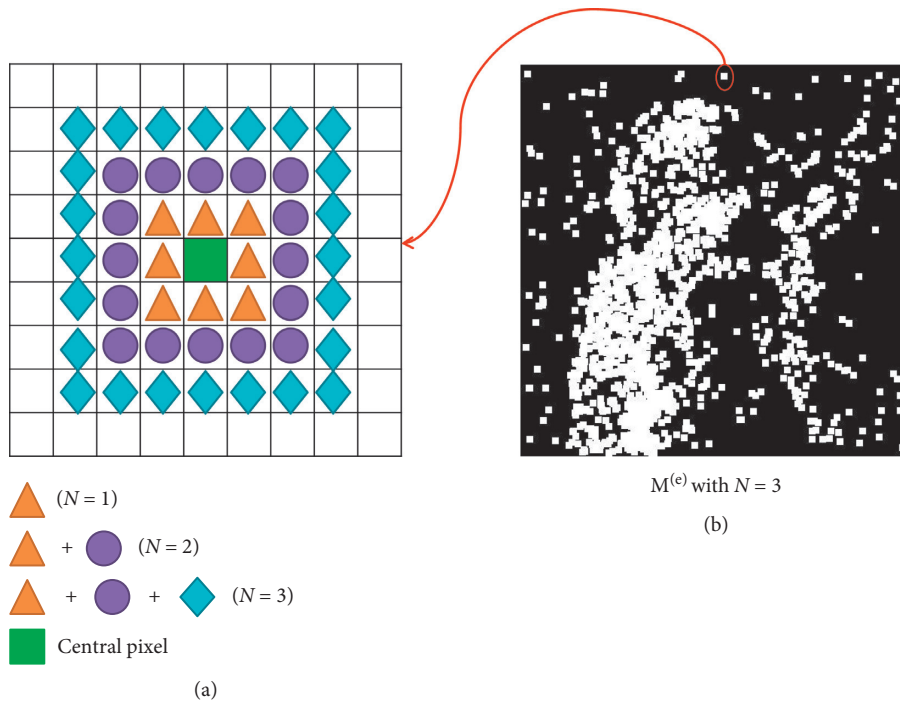


FIGURE 3: Illustration of neighboring regions of a central pixel in different margin values, and the extended modification map $\mathbf{M}^{(e)}$ with margin value $N = 3$.

number of pixels flipped by the first embedding (those pixels equal to 255 in $\mathbf{M}^{(1)}$). Besides, m denotes the number of pixels that are flipped by the first and second embedding, in which the pixels equal 255 in both $\mathbf{M}^{(1)}$ and $\mathbf{M}^{(2)}$. r denotes the ratio of the number of pixels correctly predicted by the second embedding to the number of pixels modified by the first embedding.

To evaluate the feasibility of our proposed location algorithm, let us conduct the heuristic experiments over 10000 8-bit images from the BOSSbase ver.1.01 [47]. The experimental results in variant margin value are listed in Table 3. It should be noted that bpp denotes bits per pixel for abbreviation. One can observe that, at a fixed payload, the rate r can be increased as the margin value N becomes larger. That is because the larger adjacent regions can cover more pixels

used for information hiding. Besides, in a fixed margin value N , the more the bits embedded into, the larger proportion the modified pixels can be located. In addition, as Table 3 reports, r nearly remains stable with the large N and payload. We assume that the cost value of the pixels modified by the first embedding is slightly changed (see [20] for details). That is because those pixels are merely modified by ± 1 . When we reembed the same bits into the stego image \mathbf{S} , some pixels carrying the payload might not be flipped again.

In fact, through investigating the possibility of locating a steganographic payload, we assume that the reembedding is conducted based on the known random bits used for the first embedding. However, it cannot hinder us from locating hidden bits, even the random bits are unknown or manually totally different from the first one. The results of those

TABLE 3: r statistics on p within the margin value N when reembedding same bits.

Payload α	N										
	0	1	2	3	4	5	6	7	8	9	10
0.05 bpp	0.0645	0.3441	0.5587	0.6884	0.7701	0.8239	0.8609	0.8872	0.9063	0.9205	0.9310
0.10 bpp	0.0863	0.4340	0.6655	0.7906	0.8621	0.9050	0.9318	0.9488	0.9597	0.9669	0.9717
0.20 bpp	0.1170	0.5447	0.7799	0.8875	0.9391	0.9645	0.9771	0.9836	0.9870	0.9890	0.9902
0.30 bpp	0.1427	0.6243	0.8498	0.9364	0.9701	0.9835	0.9890	0.9916	0.9929	0.9936	0.9941
0.40 bpp	0.1659	0.6877	0.8972	0.9634	0.9841	0.9910	0.9936	0.9948	0.9954	0.9958	0.9960
0.50 bpp	0.1890	0.7431	0.9314	0.9791	0.9912	0.9947	0.9961	0.9967	0.9970	0.9972	0.9973

scenarios have been exemplified in our prior work (see [46] for details). Besides, the designed extended modification map $\mathbf{M}^{(e)}$ can to some degree predict the hidden bits but also incorrectly cover the innocent pixels. Therefore, in this paper, it is of great importance that we need to further refine the proposed extended modification map.

4.3. *Refinement of Extended Modification MAP Based on Neighboring Weight Algorithm (NWA).* As our aforementioned discussion, the selection of N is a trade-off problem. Although the increased N brings a high recall ratio, the number of incorrectly located bits is also raised. For clarity, it is worth noticing that in the design of $\mathbf{M}^{(e)}$, $(2N + 1)^2$ pixels in the neighboring region (see the colored region in Figure 3 for instance) are contained, namely, labeled as the predicted hidden bits. To refine the extended modification map, we need to choose the bits, which are probably used for the first embedding. Then, let us formulate the refined modification map as

$$\mathbf{M}^{(o)} = f[\mathbf{M}^{(e)}], \quad (4)$$

where function $f[\cdot]$ refines the original extended modification map. To this end, the problem of predicting steganographic payload transfers to designing the manner of refining the extended modification map.

For simplicity, inspired by the calculation of the cost matrix ρ of stego image $\mathbf{S}^{(1)}$, we intuitively sort the cost value ρ of each pixel in ascending order. In the stage of generating a stego image $\mathbf{S}^{(1)}$, based on the calculated cost value of \mathbf{C} , the modern adaptive algorithm tries its best to embed the hidden bits into the locations carrying low-cost values, with modification as less as possible. Thus, it makes sense that we assume the cost value of each pixel from $\mathbf{S}^{(1)}$ is similar to that of \mathbf{C} . Then, the cost value ρ guides us to complete the design of the refinement function $f[\cdot]$. Specifically, K -minimum ρ is selected, corresponding to the predicted location in the extended modification map $\mathbf{M}^{(e)}$. In other words, the locations of the set $\{\rho_1, \dots, \rho_K\}$ are selected as the optimal position. As Figure 4 illustrates, a portion of the cost matrix of a natural grey-level image is extracted for a clear demonstration. In particular, when the margin value N is set as 2 in the map $\mathbf{M}^{(e)}$, only six ρ ($K = 6$) are selected for refining the extended modification map. It should be noted that if the value K equals $(2N + 1)^2 - 1$, the refined map $\mathbf{M}^{(o)}$ degenerates back to the original extended map $\mathbf{M}^{(e)}$. The effectiveness of our proposed refinement scheme will be verified in the extensive experiments.

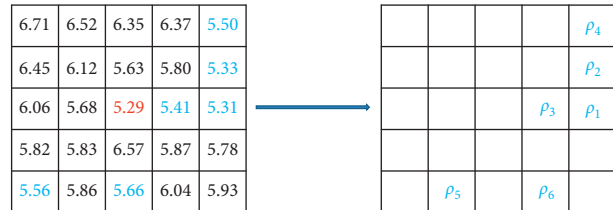


FIGURE 4: Illustration of the selection for K -minimum ρ , where $K = 6$, and the margin value $N = 2$ (at the center of value 5.29). The portion of a cost matrix (left) corresponds to the selected location in the extended modification map (right) in ascending order.

In fact, with increasing N , a labeled region where the value equals 255 (see (3)) contains more and more bits nearly irrelevant to the central pixel (labeled as 255 in the modification map $\mathbf{M}^{(o)}$), possibly leading to incorrectly predicting the hidden bits when still using the aforementioned refinement function $f[\cdot]$. For instance, the pixels located far from the central pixel carrying a low-cost value are probably selected for refinement while they have a low possibility for the first embedding. In our assumption, the hidden bits usually are embedded in the neighboring region around the central pixel. In this context, we need to consider the neighboring weight to redesign the refinement function $f[\cdot]$. Immediately, let us recalculate the cost value of each pixel by

$$\rho' = \omega \cdot \rho, \quad (5)$$

where ρ denotes the original cost value while ρ' denotes a weighted cost value, and ω denoting neighboring weight factor that is formulated by

$$\omega = \sqrt{d}, \quad (6)$$

where $d = \sqrt{(x_p - x_0)^2 + (y_p - y_0)^2}$ represents the Euclidean distance between the central pixel (x_0, y_0) and any extended pixel (x_p, y_p) , $p \in \{1, \dots, P\}$ in the extended modification map $\mathbf{M}^{(e)}$. Obviously, P is the number of pixels carrying the recalculated cost value ρ' which equals $(2N + 1)^2 - 1$. Still, among all ρ' in each map, we select the K -minimum ρ' .

For clarity, let us give an exemplary flowchart (see Figure 5) to illustrate the procedure of calculating ρ' in each extended modification map. As Figure 5 reports, although the original cost value ρ of Figure 5 is the same as that of Figure 4, the refined modification map using our proposed

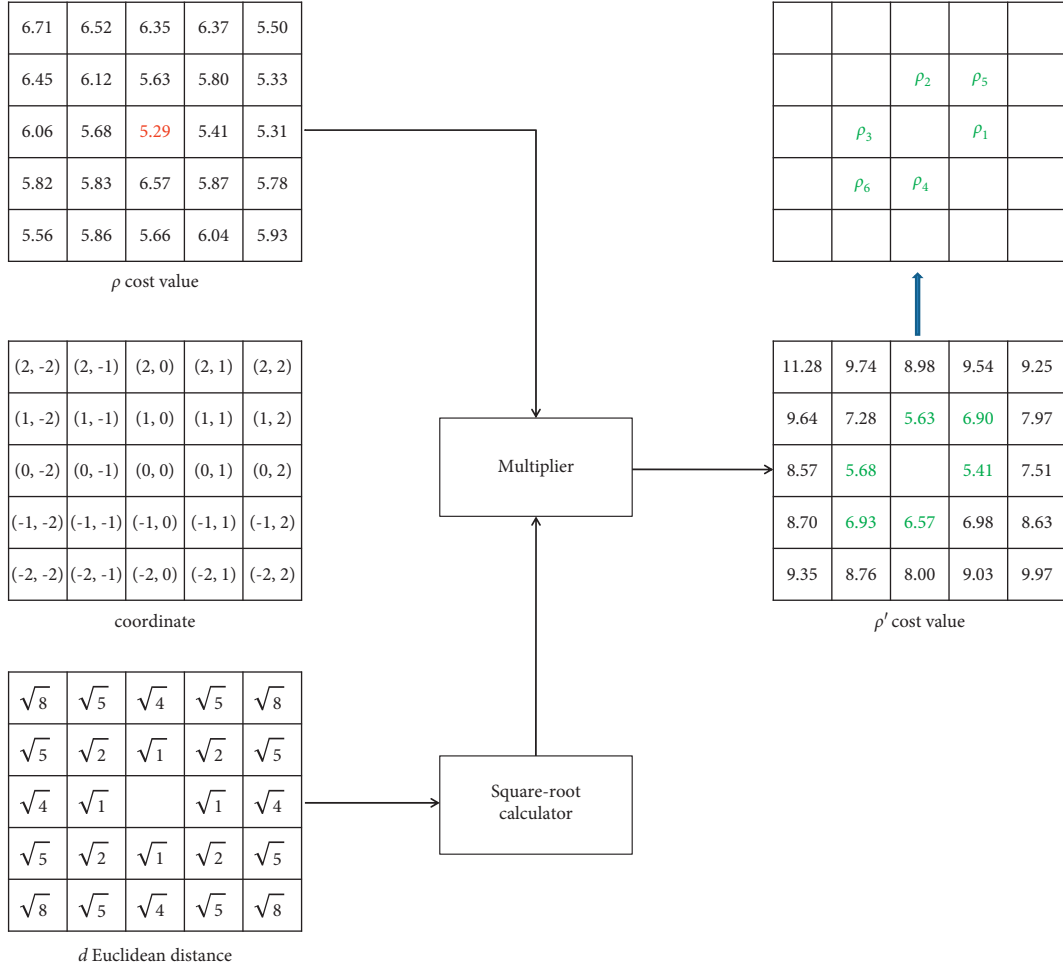


FIGURE 5: Illustration of the selection for K -minimum ρ' , where $K = 6$, and the margin value $N = 2$ (at the center of value 5.29) as Figure 4. The square-root calculator is designed based on equation (6); the multiplier is designed based on equation (5).

neighboring weight algorithm (see (6)) is different from the strategy by directly sorting the cost value ρ in ascending order (see Figure 4).

Furthermore, it is proposed to establish a more general weight factor by reformulating (6) as

$$\omega = d^n, \quad (7)$$

where n denotes the exponent of d , which is represented by

$$\begin{cases} n = 0, & \text{if } \rho \text{ is directly sorted in ascending order;} \\ n \neq 0, & \text{else.} \end{cases} \quad (8)$$

Obviously, when $n \neq 0$ holds, $n = 1/2$ represents a typical case of our redefined neighboring weight algorithm. Similarly, when $n = 0$ holds, the weight factor ω acts as a constant identity, leading to the fact that the recalculated cost value ρ' equals its original version ρ . Therefore, we propose studying the neighboring weight algorithm in the general unified framework. Then, the “square-root calculator” is replaced by a “ n -root calculator” by unifying all possible cases in our proposed framework. In the following section, we first discuss the selection of parameters n (see Section 5.2) based on the empirical experiments. Next, it is proposed to verify

the effectiveness of the proposed steganographic payload location algorithm. Finally, we compare our location algorithm with some prior arts to further validate the superiority of our algorithm.

5. Experimental Results

5.1. Experiment Setups. It is proposed to conduct numerical experiments on the baseline BOSSbase ver.1.01 [47], where all 10000 8-bit grey-level images are acquired from eight different digital still cameras in the size of 512×512 . The experimental settings are illustrated in Table 4. Besides, to comprehensively evaluate the performance of the steganographic payload location algorithm, we propose using the following metrics:

- (i) Precision $\mathcal{V}_{\mathcal{P}}$ is defined as the percentage of correctly located samples among the total number of samples (all predicted pixels containing positive and negative samples). It is formulated by

$$\mathcal{V}_{\mathcal{P}} = \frac{D_{tp}}{D_{tp} + D_{fp}}, \quad (9)$$

TABLE 4: Experimental settings.

Image source	BOSSbase ver.1.01
Image color	Grey-level
Image size	512 × 512
Image format	Uncompressed
Number of original images	10 000
Payload	0.05 ~ 0.5 bpp
Steganographic schemes	WOW, S-UNIWARD, HILL, LSBR, LSBM
Locating method	[35, 36, 38, 39, 46], ours
CPUs	4 × intel xeon E7-4820 2.0 GHz CPUs
RAM	16G

where the number of true positive samples is denoted as D_{tp} , and the number of false-positive samples (the incorrectly located pixels without flipping when embedding) is denoted as D_{fp} .

- (ii) Recall $\mathcal{V}_{\mathcal{R}}$ is the ratio of the number of samples D_{tp} to D_{tp} plus D_{fn} ; it is given by

$$\mathcal{V}_{\mathcal{R}} = \frac{D_{tp}}{D_{tp} + D_{fn}}, \quad (10)$$

where D_{fn} denotes the number of false-negative samples (the flipped pixels without being correctly located).

- (iii) F1-score $\mathcal{V}_{\mathcal{F}}$ considers both precision and recall, and it is calculated by

$$\mathcal{V}_{\mathcal{F}} = 2 \times \frac{\mathcal{V}_{\mathcal{P}} \times \mathcal{V}_{\mathcal{R}}}{\mathcal{V}_{\mathcal{P}} + \mathcal{V}_{\mathcal{R}}}. \quad (11)$$

It is worth noticing that the averaged value of each metric for all inquiry images is used to evaluate the performance of the proposed locating algorithm.

5.2. Parameter Selection of Neighboring Weight Algorithm.

In this section, we empirically verify the selection of the neighboring weight parameter for optimal location. First, it is proposed to randomly choose 1000 grey-level images from the benchmark dataset BOSSbase. Next, by adopting S-UNIWARD steganography, we embed secret bits into the cover source with a 0.3 payload. In virtue of our proposed algorithm, the weight factor ω mainly controls the cost value ρ (see (5) and (7)) for each ready-to-located pixel. Moreover, the dimension of the ready-to-located region containing both flipped and nonflipped pixels is directly decided by the parameter K . Therefore, let us empirically select the optimal parameters for the proposed locating algorithm. To comprehensively evaluate the performance of the proposed neighboring weight algorithm, we report the location results using three metrics, referring to as precision, recall, and F1-score (see Figure 6).

As Figure 6(a) illustrates, with increasing the K value, the $\mathcal{V}_{\mathcal{P}}$ is gradually falling down, meaning that more and more nonflipped (or innocent) pixels are incorrectly located. Since the large K probably generates the high-dimensional region

including innocent pixels, the locating precision is decreased when the increased number of correctly located pixels cannot match the increased number of incorrectly located pixels. Additionally, at the small K (not larger than 6), the differences of the $\mathcal{V}_{\mathcal{P}}$ using various n behave very similarly. When K equals 1, $\mathcal{V}_{\mathcal{P}}$ with $n = 0$ is remarkably better than the others. That is because the proposed algorithm carefully selects one minimum cost ρ' without considering the neighboring weight factor ω . In this scenario, it cannot hold true that the information of distance impacts the precision of payload location. On the contrary, when the K is enlarged, the performance of the locating algorithm is obviously declined. It is worth noting that the slope of $\mathcal{V}_{\mathcal{P}}$ with $n = 0$ is steeper than the others. That is because when more payloads need to be located, we intend to centralize them around the central pixel while not locating pixels with low cost possibly in the far edge (the case of $n = 0$), which are impossibly used for embedding in our assumption. Accordingly, only relying on the empirical analysis of precision $\mathcal{V}_{\mathcal{P}}$, the selection of $n = 1/4$ is capable of bringing us the optimal locating result.

In Figure 6(b), we also investigate the performance of the proposed locating algorithm by comparing the K ranging from 1 to 12 and n lying between 0 and 2. With increasing the K value, the recall $\mathcal{V}_{\mathcal{R}}$ parameterized with different weight factors is improved. In fact, when calculating the value of recall rate, the D_{fp} is not counted. In this scenario, as the dimension of locating region is enlarged, more D_{tp} is counted while ignoring the number of pixels incorrectly located. Obviously, based on the investigation of locating performance relying on the recall $\mathcal{V}_{\mathcal{R}}$, the K equal to 12, together with $n = 2$, is our optimal choice, which is totally different from the result of parameter selection based on $\mathcal{V}_{\mathcal{P}}$ (see Figure 6(a)).

Without loss of generality, the precision $\mathcal{V}_{\mathcal{P}}$ denotes the rate of locating accuracy, and the recall $\mathcal{V}_{\mathcal{R}}$ represents if all the flipped pixels are comprehensively located. To strike the balance of two metrics for ideal selection, let us demonstrate the results of the F1-score $\mathcal{V}_{\mathcal{F}}$ in Figure 6(c). As we expected, when the K approaches 4, the F1-score value basically remains stable while achieving the maximum value at K equal to 6. Meanwhile, the n equal to 1/2 is the optimal choice for our proposed locating algorithm, which will be applied in the following experiments.

5.3. Case Studies for Locating Hidden Bits. Let us first evaluate the performance of our proposed locating algorithm in four cases, which has been specifically described in Section 3.2.

5.3.1. KPKS (Known Payload and Known Scheme). 10000 cover images from BOSSbase ver.1.01 are used for generating stego images, among which we adopt well-performed S-UNIWARD and HILL steganography, respectively. To overall verify the effectiveness of the locating algorithm, it is proposed to use various payloads ranging from 0.1 to 0.5 at step 0.1. Besides, as the prior work [46], it is compared with our proposed algorithm. For simplicity and clarity, let us name the algorithm [46] as LAS (see the description in the

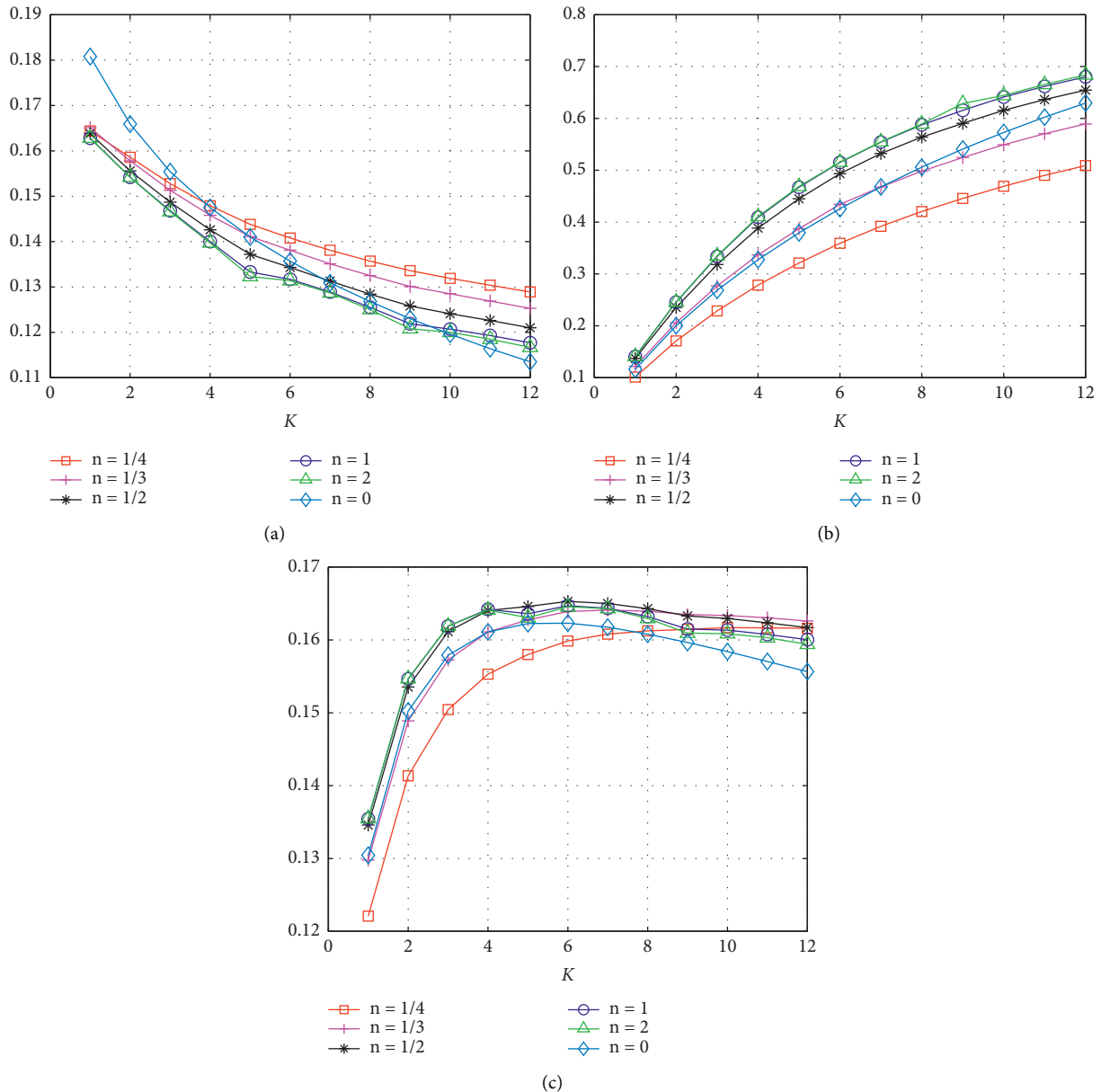


FIGURE 6: Illustration of location performance using different n , which mainly controls the general weight factor $\omega = d^n$, where d denotes the Euclidean distance; three metrics, namely, precision, recall, and F1-score, are used for evaluation: (a) precision, (b) recall, and (c) F1-score.

following section) without NWA while our proposed scheme LAS with NWA. That is because the main differences between them are whether the locating scheme is designed based on the neighboring weight algorithm (NWA).

As Figure 7 illustrates, our proposed LAS with NWA slightly performs better than that of LAS without NWA at all the given payloads. Meanwhile, with increasing the payload, the performances of both algorithms are gradually improved. That is because the large payload brings more hidden bits embedded in the region, where the extended modification map can cover. Moreover, by assigning the weight, LAS with NWA further improves the performance of locating algorithms targeting modern adaptive steganography. It should be noted that the performance gap of two

compared locating algorithms is gradually narrowed down as payload increases. That is because more hidden bits (payload 0.5 for instance) embedded into the carrier source nearly cover both texture and nontexture region, leading to the fact that the effectiveness of the selection of pixels with minimizing embedding distortion is not as remarkable as that of the small payload (a 0.1 payload for instance). In fact, when designing adaptive steganographic schemes, a similar case also happens.

Besides, by comparing S-UNIWARD with HILL, obviously, the hidden bits from stego image adopted by HILL are easier to be located. To our knowledge, the detection error of steganalysis (only targeting the problem of binary classification between the cover and stego source), referring to as

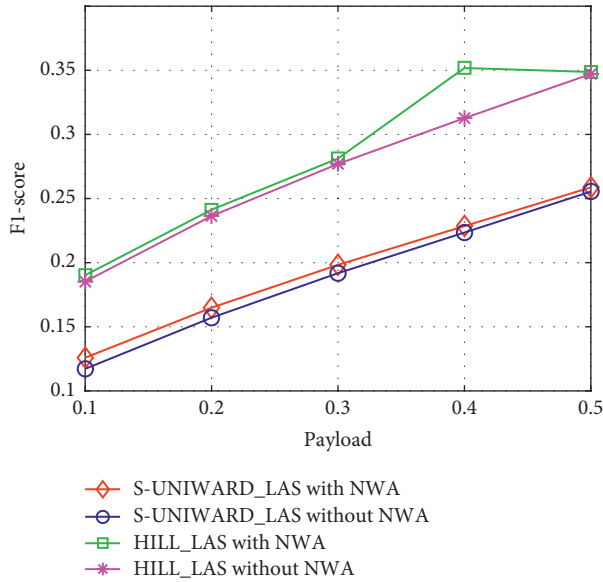


FIGURE 7: Averaged F1-score comparison between LAS without NWA [46] and our proposed LAS with NWA, in the case of KPUS.

E_{ob} or P_E , is usually adopted to evaluate the undetectability of steganography. In such a manner, HILL steganography is always regarded as the better choice than its opponent S-UNIWARD [4] while it hardly holds true that S-UNIWARD performs worse than HILL as the localization resistance of steganography is considered at the same time, which is empirically verified via our proposed locating algorithm.

5.3.2. KPUS (Known Payload and Unknown Scheme).

Practically, a steganalyzer possibly has no idea of the specific algorithm used for data hiding. In such a case, it is proposed to evaluate the performance of the proposed LAS with NWA using the cost function from different adaptive steganography methods, also compared to LAS without NWA. 10000 cover images are used for generating stego images with a 0.3 payload, among which WOW, S-UNIWARD, and HILL are adopted.

As Table 5 reports, we list 9 pairs of comparison data, where the former data corresponds to F1-score from LAS with NWA, and the latter underlined data are obtained from LAS without NWA. It can be obviously observed that our proposed LAS with NWA performs better than LAS without NWA. Basically, when predicting the embedding scheme correctly, we can acquire the larger F1-score, meaning the better location result. Even if the steganographic method is predicted incorrectly, the performance is not decreased sharply, implying that the cost function from various adaptive steganographic methods cannot serve as a decisive factor for locating hidden bits. In fact, whatever adaptive scheme is adopted, it always searches for the texture region in the image for minimizing embedding cost.

In addition, when adopting the cost function from WOW steganography, LAS with NWA performs best in not only the correctly predicted label but also the mismatched

label, S-UNIWARD for instance. That is because WOW is prone to modify pixels centralized in the regions that are difficult to model while S-UNIWARD to some extent disperses its modification for security. Nevertheless, when not knowing the adaptive scheme, it can be predicted as WOW steganography.

5.3.3. UPKS (Unknown Payload and Known Scheme).

Before locating the hidden bits, it is required to know the specific amount of payload of an inquiry image. However, in the more practical case that the payload is unknown, we have to predict it prior to locating hidden bits. Then, effective quantitative steganalysis [15] is adopted for accurately predicting the payload. To verify the effectiveness of the prediction algorithm, 4000 stego images are experimentally tested by, respectively, using S-UNIWARD and HILL steganography, in which half of them is with the payload 0.3 and half of them with the payload 0.5. Thus, the number of each type of stego images is 1000. Then, the histograms of prediction error are illustrated in Figures 8(a) and 9(a). It can be observed that the prediction error is relevantly small, where most of the data are concentrated around zero (perfectly correct prediction). Thus, the quantitative steganalysis is reliable enough, which can serve our proposed locating algorithm. Besides, with increasing payload, the overall error is narrowed down, meaning that the more payload is given, the more accurate prediction we can obtain.

Next, let us further investigate whether the hidden bits can be successfully located relying on the predicted payload. In such a case, it is proposed to compare the F1-score result of UPKS with that of KPUS serving as the baseline ground truth. When the payload is 0.3, two modern steganographic schemes are adopted. Two histograms in each figure are nearly overlapped, meaning that the F1-score of UPKS basically matches that of KPUS (see Figures 8(b) and 9(b) for details). Besides, at the payload 0.5, the comparison results are illustrated in Figures 8(c) and 9(c), respectively, which also verify the effectiveness of our proposed locating algorithm. Moreover, we calculate the statistical parameters of the histogram, referring to mean and variance values of both compared histograms. In Figure 8(c), for instance, both mean and variance values of KPUS and UPKS are equal to 0.2556 and 0.0028. Therefore, the experimental results empirically verify that thanks to the accurate prediction of payload, our proposed LAS with NWA can still work very well for locating hidden bits in the case of UPKS.

5.3.4. UPUS (Unknown Payload and Unknown Scheme).

Finally, let us evaluate the effectiveness of the proposed locating algorithm in the most difficult scenario, referring to as neither knowing payload nor specific adaptive embedding scheme. In this case, relying on the empirical analysis from KPUS and UPKS, it is proposed to first predict the payload and then locate hidden bits by using our proposed LAS with NWA based on the cost function of WOW steganography.

Also, the results of KPUS serve as the baseline for comparison. As Figure 10 illustrates, by comparing the F1-score between KPUS and UPUS, the overall result of KPUS is

TABLE 5: Averaged F1-score comparison between LAS with NWA and LAS without NWA [46] at the payload 0.3, in the case of KPUS.

True scheme	Predicted scheme		
	WOW	S-UNIWARD	HILL
WOW	0.318 5 , 0.307 6	0.231 1, 0.253 2	0.263 6, 0.272 8
S-UNIWARD	0.255 5 , 0.219 2	0.198 3, 0.191 7	0.220 9, 0.200 2
HILL	0.268 7, 0.265 1	0.196 6, 0.224 6	0.281 2 , 0.277 0

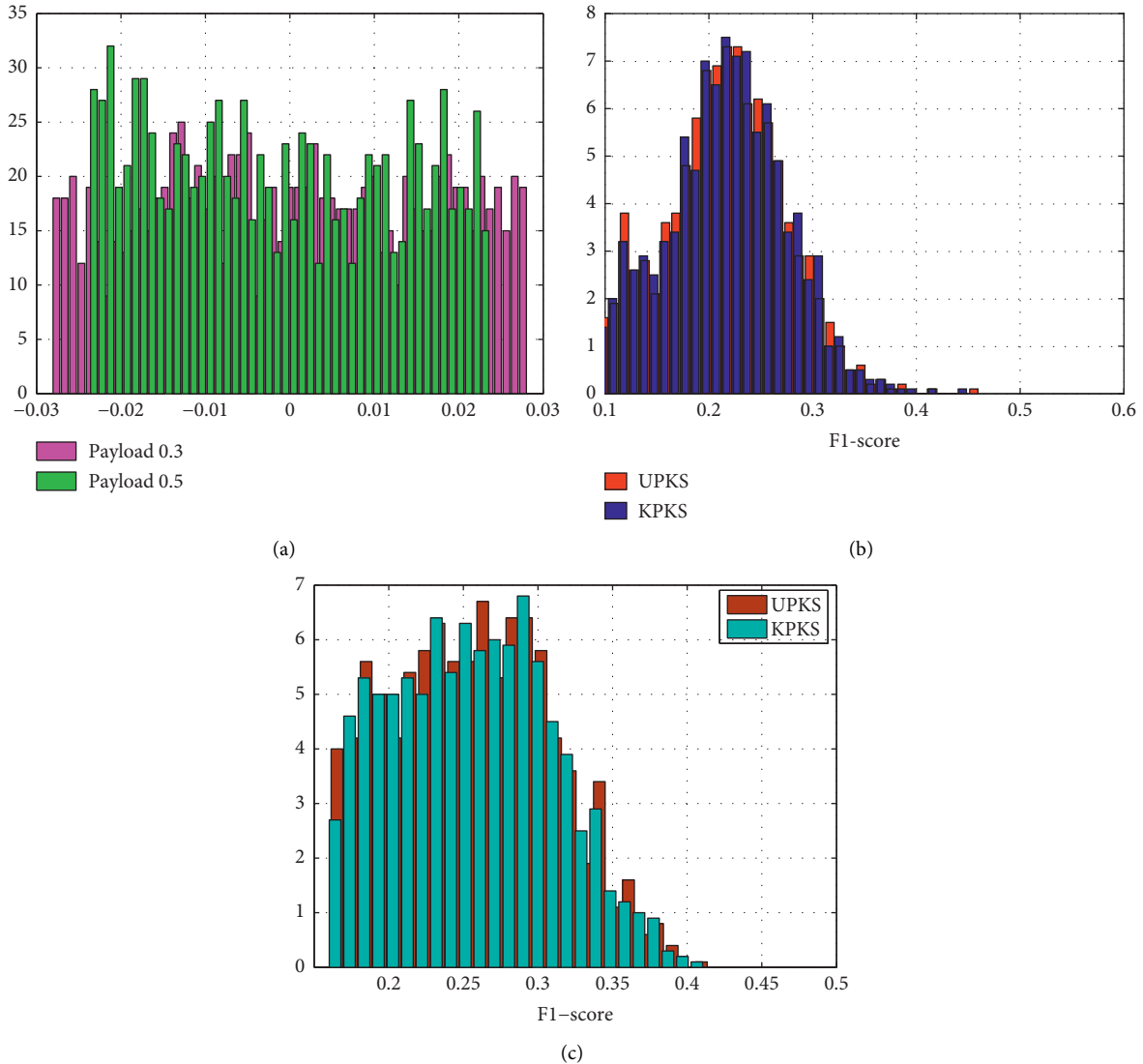


FIGURE 8: Performance of our proposed locating algorithm targeting S-UNIWARD. (a) Error histogram between the predicted payload and its ground truth. (b) Histogram of F1-score in the case UPKS and KPKS at the payload 0.3. (c) Histogram of F1-score in the case UPKS and KPKS at the payload 0.5.

obviously superior to that of UPUS, especially at the large-value bin of histogram, meaning that the performance of locating is slightly degraded in the case UPUS. Moreover, the error histogram is also illustrated by making differences between F1-scores of two cases (the results of KPUS minus that of UPUS). As Figure 11 reports, most of the data larger than zero directly validates the better performance of the locating algorithm in the case KPUS. Lack of enough information about a specific amount of payload and

embedding scheme unavoidably leads to the fact that the extended modification map is hardly constructed, which more or less impacts the accuracy of locating hidden bits.

5.4. Comparison with Prior Arts. Compared with the baseline prior arts, the superiority of our proposed locating algorithm is experimentally verified. For simplicity and clarity, let us describe the prior arts, referring to as 5 algorithms

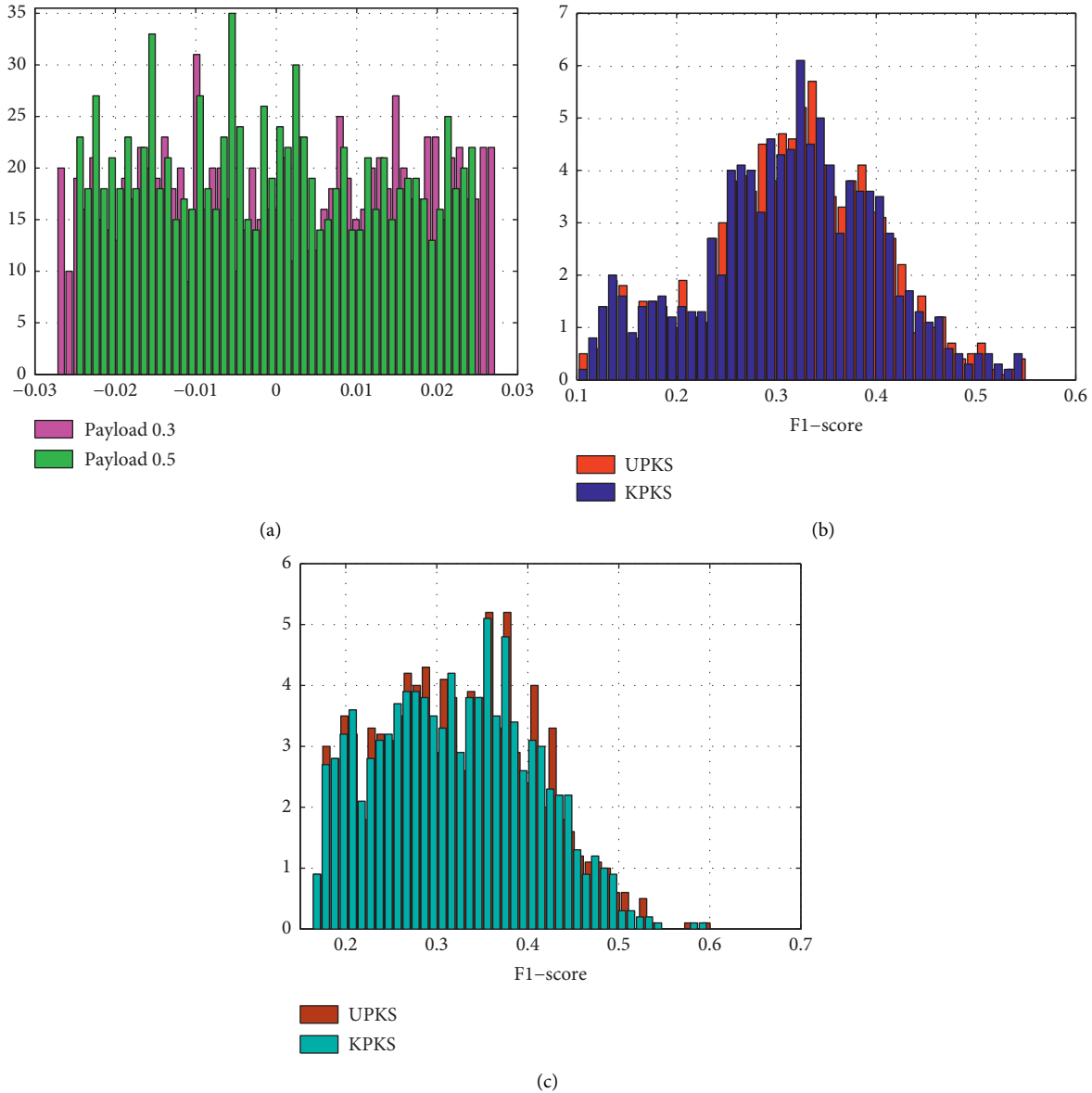


FIGURE 9: Performance of our proposed locating algorithm targeting HILL. (a) Error histogram between the predicted payload and ground truth. (b) Histogram of F1-score in the case UPKS and KPKS at the payload 0.3. (c) Histogram of F1-score in the case UPKS and KPKS at the payload 0.5.

[35, 36, 38, 39, 42] toward nonadaptive steganography, a novel algorithm [46] toward adaptive steganography, and adaptive steganalysis (not originally designed for locating hidden bits) [48, 49]. Each algorithm is elaborated as follows.

WSR [35]: by using the linear filter, each cover pixel can be approximately estimated. Then, each residual noise is calculated by making the differences between the stego pixel and its estimated cover one. The stego pixel carrying hidden bit is located by comparing the averaged residual noise with the preset threshold, such as 0.25 for instance. Furthermore, by assigning weight to residual noise, the performance of the Weighted Stego Residual (WSR) algorithm is improved. The limitation of it is that the secret key for each image should remain unchanged; it is designed only for LSBR. WAM [36]: relying on an 8-tap Daubechies kernel, pixels in the spatial

domain are converted to coefficients in the wavelet domain. After removing low-frequency coefficients (corresponding to subband **LL**), the remaining residual coefficients in subbands **LH**, **HL**, and **HH** are required by adopting Wavelet Absolute Moment (WAM) filter. Similar to WAR, the inversely converted residual noise in the spatial domain is used for location by comparing its magnitude with the preset threshold. The limitation of WAM is that all possible stego images share the same secret key; it is designed only for LSBM. MAP [38]: dependent on the theory Maximum A Posteriori, together with the Viterbi algorithm, the estimation of cover pixels is optimized. Similar to WSR, the residual noise is calculated between stego and cover source. Like WAM and WAS, the limitation of it is that all hidden bits are embedded in the same position for all stego images.

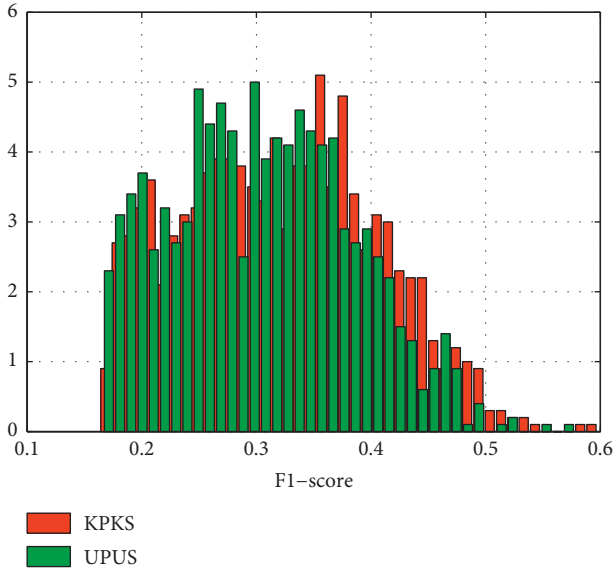


FIGURE 10: Histogram of F1-score comparison between LAS with NWA in the case UPUS and the baseline KPKS.

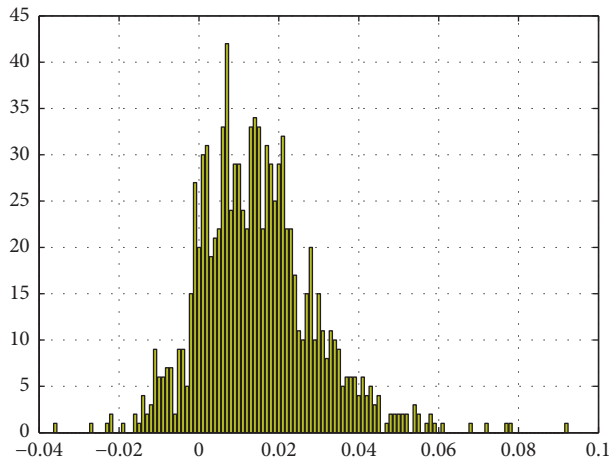


FIGURE 11: Error histogram of F1-score between LAS with NWA in the case of UPUS and the baseline KPKS.

It is worth noticing that MAP is available for both LSBR and LSBM. MRF [39]: in virtue of the Markov Random Field (MRF), a cover image is predicted by using a given stego one. In particular, dependent on pairwise constraints, the statistical features of a cover image are captured. Then, the designed locator is well performed targeting both LSBR and LSBM. DNN [42]: by taking the problem of payload location as binary classification, relying on the trained model, each pixel is treated as the predicted sample (carrying hidden bit or not), where the mean square of neighboring pixel differences serves as the key element for feature extraction. Moreover, the hand-crafted features are fed into the well-designed DNN for training an efficient model, which is available for both LSBR and LSBM. LAS [46]: by reembedding the bits into the stego image, the modification map is obtained. With the help of embedding cost, the extended version of the modification map guides us to locate the

flipped bits in the stego image. The strength of this locating algorithm is to directly target adaptive steganography (LAS), which is totally different from prior arts such as WSR, WAM, WAP, or MRF. Hu’s method [48] and Tang’s method [49]: these two methods were originally designed for image steganalysis, where the regions [48] or bits [49] with high embedding probability are preferably selected for training an efficient classifier. Thus, we insist on conducting comparison experiments with them.

Also, 10000 grey-level images from BOSSbase ver.1.01 [47] are used for comparing the performance of different locating algorithms. Here, two payloads 0.3 and 0.5 are used to generate the stego images. To enrich the experimental data, it is proposed to adopt both modern adaptive steganography and old nonadaptive steganography. It is worth noting that the number of pixels with hidden bits is fixed when LSBR or LSBM is adopted. For instance, 78643 locations need to be predicted in a 512×512 stego image with a 0.3 payload. It should be noted that MAP [38] and MRF [39] are supervised algorithms, which need to construct the trained model prior to locating hidden bits. Thus, in that case, at the given payload, half the number of images is used for training; another half is used for testing while the remaining algorithms are training-free. For a fair comparison, we should ensure that all the same 5000 images with the same payload are used for locating. Still, the F1-score serves as the comparison metric for evaluating the performance of the locating algorithm.

As Table 6 illustrates, in the case of payload 0.3, WSR is good at locating hidden bits embedded by LSBR while not LSBM. On the contrary, WAM performs very well when LSBM is used for embedding. Those results perfectly match those of [35, 36]. For supervised locating algorithms, MAP cannot only locate secret bits hidden by LSBR but also by LSBM. In addition, DNN performs very close to MAP. However, MRF cannot perform very well. The default setting of the MRF model parameter ω_1 equals 0.9986, which is acquired from the database BOSSbase ver.0.92 of [39]. That probably leads to unsatisfying results. When the payload is increased, 0.5 for instance, the performance of MRF can be further improved, which nearly matches the results of [39]. All the hidden bits embedded by LSBR or LSBM nearly can both be located (see Table 7 for details). That empirically verifies that with increasing the payload, the impact of the inaccurate MRF model parameter is able to be mitigated.

Moreover, it is noticeable that the aforementioned locating algorithms [35, 36, 38, 39, 42] only work when the stego images with hidden bits are embedded in the same position in the spatial domain. The strong assumption largely limits the extension to hidden bits location of adaptive steganography. Since modern adaptive steganography prefers to embed bits relying on the content of the cover image, it hardly holds true that the pixels in the same positions are used for embedding toward different cover images. Thus, the performance of locating algorithms [35, 36, 38, 39, 42] targeting modern steganography is not illustrated. For clarity, we utilize the notation “/” in Tables 6 and 7 denoting the invalid results. However, when targeting old steganography, our proposed LAS algorithms and two

TABLE 6: F1-score comparison of locating performance from different steganalysis methods, using both modern adaptive steganography (WOW, S-UNIWARD, and HILL) and old nonadaptive steganography (LSBR and LSBM) at the given payload 0.3.

Steganalysis locating method, steganography	WOW	S-UNIWARD	HILL	LSBR	LSBM
WSR [35]	/	/	/	1.000 0	0.280 5
WAM [36]	/	/	/	0.304 2	1.000 0
MAP [38]	/	/	/	1.000 0	1.000 0
MRF [39]	/	/	/	0.625 2	0.803 1
DNN [42]	/	/	/	0.946 4	0.941 2
Hu's method [48]	0.124 9	0.105 9	0.126 5	/	/
Tang's method [49]	0.221 4	0.172 2	0.208 5	/	/
LAS without NWA [46]	0.307 6	0.191 8	0.277 0	/	/
LAS with NWA (ours)	0.318 4	0.198 3	0.281 2	/	/

TABLE 7: F1-score comparison of locating performance from different steganalysis methods, using both modern adaptive steganography (WOW, S-UNIWARD, and HILL) and old nonadaptive steganography (LSBR and LSBM) at the given payload 0.5.

Steganalysis locating method, steganography	WOW	S-UNIWARD	HILL	LSBR	LSBM
WSR [35]	/	/	/	1.000 0	0.808 1
WAM [36]	/	/	/	0.498 4	1.000 0
MAP [38]	/	/	/	1.000 0	1.000 0
MRF [39]	/	/	/	0.988 9	0.963 6
DNN [42]	/	/	/	0.941 8	0.936 8
Hu's method [48]	0.193 7	0.167 2	0.195 3	/	/
Tang's method [49]	0.356 4	0.276 7	0.338 0	/	/
LAS without NWA [46]	0.372 5	0.255 4	0.347 1	/	/
LAS with NWA (ours)	0.375 3	0.258 7	0.348 7	/	/

adaptive steganalysis types [48, 49] fail. Due to the fact that the hidden bits are embedded randomly, the methods designed by the characteristic of modern steganography become invalid when dealing with LSBR or LSBM. Nevertheless, as Table 6 reports, when locating secret bits hidden by modern steganography, our proposed LAS with NWA outperforms the prior arts.

Additionally, we illustrate the F1-score performance of locating algorithms at the given payload 0.5 in Table 7. As we expected, whatever modern or old steganography is adopted, the performance of locating algorithms is improved compared to the results in Table 6. In fact, when more secret bits are embedded into the cover image, more location hints caused by bit modification can be provided, which definitely results in better detection. It is worth noting that our proposed LAS with NWA performs better than the others when dealing with WOW and HILL and slightly worse than Tang's method [49] when dealing with S-UNIWARD.

6. Conclusion and Limitation

In this paper, we address the problem of locating the hidden bits embedded by modern adaptive steganography. In virtue of the intrinsic property of adaptive steganography, through reembedding secret bits into stego images, we acquire the modification map. Next, based on the extended modification map, together with the neighboring weight algorithm (NWA), the location of hidden bits is further refined, leading to better performance. More importantly, for practical use, we verify the effectiveness of locating hidden bits in the four

possible cases. Prior to our study, most literature focused on locating hidden bits embedded by old steganography while ignoring the research of modern adaptive steganography. Meanwhile, a strong assumption should be given, referring to as secret bits embedded in the same position in the spatial domain for many stego images while, in our locating algorithm, only one single stego image is enough to be used for locating hidden bits.

The main limitation of the proposed algorithm is that the predicted flipped bits should be embedded by modern adaptive steganography. In other words, it fails when the old steganographic algorithm is adopted. When comparing the F1-score, we have to admit that the location accuracy of our proposed algorithm is not as good as that of the algorithms specialized in targeting old steganography (see Tables 6 and 7). In further study, we need to further improve the location accuracy targeting adaptive steganography.

Additionally, in the more generalized framework of steganalysis, on the one hand, the steganalyzer usually passively completes the task of binary classification (cover versus stego source), the amount of payload prediction (quantitative steganalysis), payload location, and hidden bits extraction (forensic steganalysis); on the other hand, he/she can also adopt the strategy of actively attacking towards steganography [50], such as interruption of covert communication or disturbing the stego carrier. However, the active disturbance is possibly nontargeted, leading to the fact that if the disturbance is too strong, referring to as randomly adding noise to overwrite the hidden bits in the stego image, for instance, the distortion of stego carrier is not acceptable;

on the contrary, too weak disturbance can hardly achieve the task of active attack.

Nevertheless, in this context, our proposed algorithm raises the promising study of payload location targeting modern adaptive steganography. Although the locations of hidden bits are not very accurately predicted, the modification region caused by embedding can be accurately located, which can indeed further help the steganalyzer actively and purposely disturb the stego image over the targeted region carrying hidden bits while mitigating the distortion caused by additional noise. Thus, it is of great importance that further steps are taken to achieve the goal of active steganalysis. Besides, we can also extend the proposed locating method to the adaptive steganalysis instead of overall feature extraction, such as [48, 49], whose effectiveness has been verified in two references, namely, channel-aware or channel-selection steganalysis for more accurate detection.

Data Availability

Data are available on request to the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the Fundamental Research Funds for the Provincial Universities of Zhejiang under grant no. GK219909299001-007, the National Natural Science Foundation of China (Grant nos. U1804263 and 62172435), and the Zhongyuan Science and Technology Innovation Leading Talent Project of China (Grant no. 214200510019).

References

- [1] V. Holub, J. Fridrich, and T. Denemark, "Universal distortion function for steganography in an arbitrary domain," *EURASIP Journal on Information Security*, vol. 2014, no. 1, p. 1, 2014.
- [2] T. Filler and J. Fridrich, "Gibbs construction in steganography," *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 4, pp. 705–720, 2010.
- [3] V. Holub and J. Fridrich, "Designing steganographic distortion using directional filters," in *Proceedings of the 2012 IEEE International workshop on information forensics and security (WIFS)*, pp. 234–239, IEEE, Tenerife, Spain, December 2012.
- [4] B. Li, M. Wang, J. Huang, and X. Li, "A new cost function for spatial image steganography," in *Proceedings of the Image Processing (ICIP), 2014 IEEE International Conference on*, IEEE, pp. 4206–4210, Paris, France, October 2014.
- [5] L. Linjie Guo, J. Jiangqun Ni, and Y. Q. Yun Qing Shi, "Uniform embedding for efficient jpeg steganography," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 5, pp. 814–825, 2014.
- [6] L. Guo, J. Ni, W. Su, C. Tang, and Y.-Q. Shi, "Using statistical image model for jpeg steganography: uniform embedding revisited," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 12, pp. 2669–2680, 2015.
- [7] Z. Zhao, Q. Guan, H. Zhang, and X. Zhao, "Improving the robustness of adaptive steganographic algorithms based on transport channel matching," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 7, pp. 1843–1856, 2018.
- [8] J. Tao, S. Li, X. Zhang, and Z. Wang, "Towards robust image steganography," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 2, pp. 594–600, 2019.
- [9] Y. Zhang, X. Luo, Y. Guo, C. Qin, and F. Liu, "Multiple robustness enhancements for image adaptive steganography in lossy channels," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 8, pp. 2750–2764, 2019.
- [10] T. Qiao, S. Wang, X. Luo, and Z. Zhu, "Robust steganography resisting jpeg compression by improving selection of cover element," *Signal Processing*, vol. 183, Article ID 108048, 2021.
- [11] T. Qiao, C. Zitzmann, F. Retraint, and R. Cogranne, "Statistical detection of jsteg steganography using hypothesis testing theory," in *Proceedings of the Image Processing (ICIP), 2014 IEEE International Conference on*, pp. 5517–5521, IEEE, Paris, France, October 2014.
- [12] T. Qiao, C. Zitzmann, R. Cogranne, and F. Retraint, "Detection of jsteg algorithm using hypothesis testing theory and a statistical model with nuisance parameters," in *Proceedings of the 2nd ACM workshop on Information hiding and multimedia security*, pp. 3–13, ACM, Salzburg, Austria, June 2014.
- [13] T. Qiao, F. Retraint, R. Cogranne, and C. Zitzmann, "Steganalysis of jsteg algorithm using hypothesis testing theory," *EURASIP Journal on Information Security*, vol. 2015, no. 1, p. 2, 2015.
- [14] T. Qiao, X. Luo, T. Wu, M. Xu, and Z. Qian, "Adaptive steganalysis based on statistical model of quantized dct coefficients for jpeg images," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 6, pp. 2736–2751, 2019.
- [15] T. Pevny, J. Fridrich, and A. D. Ker, "From blind to quantitative steganalysis," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 445–454, 2012.
- [16] J. Fridrich, M. Goljan, and D. Soukal, "Searching for the stego-key," *Security, Steganography, and Watermarking of Multimedia Contents VI*, vol. 5306, pp. 70–83, 2004.
- [17] J. Liu, Y. Tian, T. Han, J. Wang, and X. Luo, "Stego key searching for lsb steganography on jpeg decompressed image," *Science China Information Sciences*, vol. 59, no. 3, Article ID 32105, 2016.
- [18] C. Xu, J. Liu, J. Gan, and X. Luo, "Stego key recovery based on the optimal hypothesis test," *Multimedia Tools and Applications*, vol. 77, no. 14, pp. 17973–17992, 2018.
- [19] J. Kodovský, J. J. Fridrich, and V. Holub, "Ensemble classifiers for steganalysis of digital media," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 2, pp. 432–444, 2012.
- [20] W. Tang, H. Li, W. Luo, and J. Huang, "Adaptive steganalysis based on embedding probabilities of pixels," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 4, pp. 734–745, 2016.
- [21] Y. Ma, X. Luo, X. Li, Z. Bao, and Y. Zhang, "Selection of rich model steganalysis features based on decision rough set α -positive region reduction," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 2, pp. 336–350, 2019.
- [22] Z. Wang, Z. Qian, X. Zhang, and S. Li, "An improved steganalysis method using feature combinations," in *Proceedings of the International Conference on Artificial Intelligence and*

- Security*, pp. 115–127, Springer, New York, NY, USA, July 2019.
- [23] B. Chen, W. Tan, G. Coatrieux, Y. Zheng, and Y. Q. Shi, “A serial image copy-move forgery localization scheme with source/target distinguishment,” *IEEE Transactions on Multimedia*.
- [24] B. Chen, X. Liu, Y. Zheng, G. Zhao, and Y.-Q. Shi, *A Robust gan-generated Face Detection Method Based on Dual-Color Spaces and an Improved Xception*, IEEE Transactions on Circuits and Systems for Video Technology, New York, NY, USA.
- [25] J. Ye, J. Ni, and Y. Yi, “Deep learning hierarchical representations for image steganalysis,” *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 11, pp. 2545–2557, 2017.
- [26] W. Tang, B. Li, M. Barni, J. Li, and J. Huang, “An automatic cost learning framework for image steganography using deep reinforcement learning,” *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 952–967, 2020.
- [27] W. Tang, B. Li, M. Barni, J. Li, and J. Huang, “Improving cost learning for jpeg steganography by exploiting jpeg domain knowledge,” <https://arxiv.org/abs/2105.03867>.
- [28] J. Butora, Y. Yousfi, and J. Fridrich, “How to pretrain for steganalysis,” in *Proceedings of the 2021 ACM Workshop on Information Hiding and Multimedia Security*, pp. 143–148, Brussels, Belgium, July 2021.
- [29] Y. Yousfi, J. Butora, J. Fridrich, and C. Fuji Tsang, “Improving efficientnet for jpeg steganalysis,” in *Proceedings of the 2021 ACM Workshop on Information Hiding and Multimedia Security*, pp. 149–157, Brussels, Belgium, July 2021.
- [30] W. You, H. Zhang, and X. Zhao, “A siamese cnn for image steganalysis,” *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 291–306, 2020.
- [31] W. Ren, L. Zhai, J. Jia, L. Wang, and L. Zhang, “Learning selection channels for image steganalysis in spatial domain,” *Neurocomputing*, vol. 401, pp. 78–90, 2020.
- [32] J. Zhang, K. Chen, C. Qin, W. Zhang, and N.-H. Yu, “Distribution-preserving-based automatic data augmentation for deep image steganalysis,” *IEEE Transactions on Multimedia*, pp. 1–13, 2021.
- [33] H. Yang, H. He, W. Zhang, and X. Cao, “Fedsteg: a federated transfer learning framework for secure image steganalysis,” *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 2, pp. 1084–1094, 2020.
- [34] M. Chen, M. Boroumand, and J. Fridrich, “Deep learning regressors for quantitative steganalysis,” *Electronic Imaging*, vol. 7, pp. 1–7, 2018.
- [35] A. D. Ker, “Locating steganographic payload via ws residuals,” in *Proceedings of the 10th ACM workshop on Multimedia and security*, pp. 27–32, ACM, Oxford, UK, August 2008.
- [36] A. D. Ker and I. Lubenko, “Feature reduction and payload location with wam steganalysis,” *Media forensics and security*, vol. 7254, Article ID 72540A, 2009.
- [37] Y. Luo, X. Li, and B. Yang, “Locating steganographic payload for lsb matching embedding,” in *Proceedings of the 2011 IEEE International Conference on Multimedia and Expo*, pp. 1–6, IEEE, Barcelona, Spain, July 2011.
- [38] T.-T. Quach, “Optimal cover estimation methods and steganographic payload location,” *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 4, pp. 1214–1222, 2011.
- [39] T.-T. Quach, “Cover estimation and payload location using Markov random fields,” *Media Watermarking, Security, and Forensics 2014*, vol. 9028, Article ID 90280H, 2014.
- [40] T. Pevny, P. Bas, and J. Fridrich, “Steganalysis by subtractive pixel adjacency matrix,” *IEEE Transactions on Information Forensics and Security*, vol. 5, no. 2, pp. 215–224, 2010.
- [41] X. Yan, T. Zhang, L. Xi, and X. Ping, “New method for payload location aimed at lsb matching,” *Journal of Data Acquisition & Processing*, vol. 31, no. 1, pp. 145–151, 2016.
- [42] Y. Sun, H. Zhang, T. Zhang, and R. Wang, “Deep neural networks for efficient steganographic payload location,” *Journal of Real-Time Image Processing*, vol. 16, no. 3, pp. 635–647, 2019.
- [43] J. Wang, C. Yang, P. Wang, X. Song, and J. Lu, “Payload location for jpeg image steganography based on co-frequency sub-image filtering,” *International Journal of Distributed Sensor Networks*, vol. 16, no. 1, Article ID 1550147719899569, 2020.
- [44] J. Wang, C. Yang, M. Zhu, X. Song, Y. Liu, and Y. Lian, “Jpeg image steganography payload location based on optimal estimation of cover co-frequency sub-image,” *EURASIP Journal on Image and Video Processing*, vol. 2021, no. 1, pp. 1–14, 2021.
- [45] J. Liu, C. Yang, J. Wang, and Y. Shi, “Stego key recovery method for f5 steganography with matrix encoding,” *EURASIP Journal on Image and Video Processing*, vol. 2020, no. 1, pp. 1–17, 2020.
- [46] Q. Liu, T. Qiao, M. Xu, and N. Zheng, “Fuzzy localization of steganographic flipped bits via modification map,” *IEEE Access*, vol. 7, pp. 74157–74167, 2019.
- [47] P. Bas, T. Filler, and T. Pevný, ““Break our steganographic system”: the ins and outs of organizing BOSS,” in *Information Hiding* Springer, New York, NY, USA, 2011.
- [48] D. Hu, Q. Shen, S. Zhou, X. Liu, Y. Fan, and L. Wang, “Adaptive steganalysis based on selection region and combined convolutional neural networks,” *Security and Communication Networks*, vol. 20179 pages, 2017.
- [49] W. Tang, H. Li, W. Luo, and J. Huang, “Adaptive steganalysis against wow embedding algorithm,” in *Proceedings of the 2nd ACM workshop on Information hiding and multimedia security*, ACM, pp. 91–96, 2014.
- [50] J. M. Ettinger, “Steganalysis and game equilibria,” in *International Workshop on Information Hiding* Springer, New York, NY, USA, 1998.

Research Article

Face Forgery Detection Based on the Improved Siamese Network

Bo Wang ¹, Yucai Li ¹, Xiaohan Wu ¹, Yanyan Ma ¹, Zengren Song ²,
and Mingkan Wu ¹

¹School of Information and Communication Engineering, Dalian University of Technology, Dalian, China

²National Computer Network Emergency Response Technical Team/Coordination Center of China, Beijing, China

Correspondence should be addressed to Yucai Li; lyc124184@mail.dlut.edu.cn

Received 30 November 2021; Revised 28 December 2021; Accepted 8 January 2022; Published 5 February 2022

Academic Editor: Beijing Chen

Copyright © 2022 Bo Wang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Face tampering is an intriguing task in video/image genuineness identification and has attracted significant amounts of attention in recent years. In this work, we propose a face forgery detection method that consists of preprocessing, an improved Siamese network-based feature extractor (including a feature alignment module), and postprocessing (a voting principle). Roughly speaking, our method extracts the features in the grey space of face/background image pairs and measures the difference to make decisions. Experiments on several standard databases prove the effectiveness of our method, and especially on the low-quality subdataset of the FaceForensics++ , our method achieves a competitive result.

1. Introduction

In recent years, image/video tampering methods have developed rapidly [1], including Deepfake [2], Face2Face [3], FaceSwap [4], and Neural Textures [5]. These methods rely on advanced image/video processing algorithms and are embedded within many applications in the market. Because visual contents can be easily manipulated, the detection of tampered contents is of practical significance and readily attracts attention [6]. In this work, we are interested in face forgery detection.

Many methods have been proposed for detection of tampered face images and videos, and the accuracy mainly depends on the selection of features and classifiers. The state-of-the-art methods roughly consist of two stages: feature extraction and classification. Several methods segregate these stages as separate subproblems [7–10], while some methods integrate the two stages in sequence based on deep neural networks (DNNs) [1, 11–18]. Regarding face forgery detection, there are two main types of selections of features: one is based on single-image features [1, 7–29], while the other is based on between-frame feature differences in videos [30–39]. Note that various types of classifiers are used (e.g., SVM, CNN, RNN, and MLP) and that SVM and CNN are relatively more popular.

The existing methods have achieved excellent detection accuracy on public datasets, including [1, 23, 40–43]. However, there are still problems yet to be solved. The first problem is that most methods offer poor robustness. They can achieve satisfactory accuracy on uncompressed or lightly compressed images and videos, but for content that is compressed with high intensity, the detection accuracy is greatly reduced because the compression may significantly eliminate the traces of forgery. The quality of images and videos also decreases after rounds of postprocessing, which greatly compromises the performance of the existing methods. The second problem is that almost all of the methods use only the features of the facial area or the fusion boundary area of the face and background but discard the features of the background. Although normally only the facial area is tampered, it is worth noting that, for untampered images, the facial area and the background are consistent at a certain feature level, which stands in contrast with forged images. Therefore, in this work, we address the face-background difference-based features.

In this paper, we propose a method based on the improved Siamese network [44]. The Siamese network was originally proposed to learn a similarity metric with application to face verification. We use the Siamese network to

measure the similarity between the face area and the background of the video frames. Before being saved in memory, a captured video is processed through a series of steps, including quantization, denoising, color correction, gamma correction, filtering, white balance, and even JPEG compression [45]. This series of processing steps involves unique statistical characteristics, and in an untampered video, the face area and the background of the video frames exhibit high similarity. In a tampered video, the similarity between the face area and the background is low because they originate from different videos. It is worth noting that this specialty is video-level; that is, the similarity relationship between the face area and the background between different frames in the video conforms to this law, because all processing is carried out on the whole video, that is, all frames. Our improved Siamese network can measure the similarity in order to distinguish genuine and tampered images and videos. The general pipeline of our method is depicted in Figure 1, and our contribution can be roughly concluded as follows: First, we design a preprocessing module that obtains a large number of image patch pairs of face area and background. Next, we present our improved Siamese network, which consists of two submodules, i.e., feature extraction and feature alignment. In the feature extraction module, we grey the image patch pairs and then input the pairs to a two-stream convolutional neural network with shared weights to extract features in the grey space of the images. In the feature alignment module, we align the features to measure the similarity between the face area and the background of images and obtain the final authenticity judgement result. During testing, we define a voting principle to correct our results by cropping multiple pairs of face area and background from a video frame. Then, we define a voting principle to correct the classification results. Last, through experiments, we show that our method outperforms the state-of-the-art methods on challenging low-quality datasets.

2. Related Work

2.1. Face Forgery. The most widely used face tampering methods include Deepfake [2], Face2Face [3], FaceSwap [4], and Neural Textures [5]. Examples of these methods are depicted in Figure 2.

The core of the application of Deepfake to facial video tampering is the parallel training of two autoencoders with shared parameters. The production process has two stages: the training stage and the generation stage. In the training stage, two autoencoders with shared parameters extract the features of two faces that belong to different persons and then input two autoencoders with independent parameters. In the generation stage, the facial features extracted by the autoencoder are input into the autoencoder corresponding to another different face to generate a mixed face. Finally, the mixed face is blended with the rest of the image using Poisson image editing [46]. Face2Face is a technology that can modify the expression and mouth shape of the target character. The main advancement of Face2Face lies in deforming various algorithms, including improvements in RGB tracking

algorithms, transfer functions, and the establishment of mouth models. FaceSwap is used to transfer the face area from the source video to the target video. For the source video, the method first extracts the facial area of the source video and its corresponding facial landmarks and then fits a 3D model. For the target video, the method uses the same approach to fit the 3D model, which is rendered by the texture coordinates obtained from the 3D model of the source video to produce the final face-changing video. Neural Textures uses expression migration to modify the texture map of the target actor's face to match the expression of the source actor. This texture map is used to sample the neural texture of the target character. Then, the method inputs the sampled neural texture map to the delayed neural renderer and outputs the final reproduction result after end-to-end training.

2.2. Detection of Face Forgery. With the development of face tampering technology, the forged images and videos produced are close to genuine, which has aroused concerns and attracted attention to research on detection technology for face tampering. Existing detection methods can be roughly divided into two types: detection for tampered images and videos.

2.2.1. Detection Methods for Tampered Images. This type of method aims to extract the features of the single image for classification. Some traditional manual features such as speeded up robust features (SURF) [7], photo response nonuniformity (PRNU) [8], local binary pattern (LBP) [9], image quality measures (IQM) [10], etc., can be used to detect tampered images. However, the accuracy of these methods is not competitive on large datasets. With the rapid development of deep learning, face forgery detection has also made extensive use of deep learning. Deep neural networks (DNN) are used to extract the features of a single image or as classifiers. Some methods use DNN to extract the frequency features of the images [19–22]. For example, Luo et al. [20] found that current CNN-based detectors tend to overfit to method-specific color textures and thus fail to generalize, so they proposed to utilize the high-frequency noises for face forgery detection by devising three functional modules observing image noises remove color textures and expose discrepancies between authentic and tampered regions. Besides, the unique biological features of face images are used as classification features by some methods [23–26]. Matern et al. [24] proposed a method to detect Deepfake videos based on the visual features of eyes, teeth, and facial contours. However, this method has certain requirements for the test images, such as that the images need to include clear eyes or teeth. References [27–29] effectively used the texture or boundary features of the images and had a certain improvement in cross-database detection performance. There are some methods that use specific neural networks to detect tampered images with end-to-end training [1, 11–15, 39] and some methods [16–18] also introduce attention mechanism on this basis. These methods rely on the powerful adaptive learning ability of the neural network and the focus of the methods is therefore on the construction

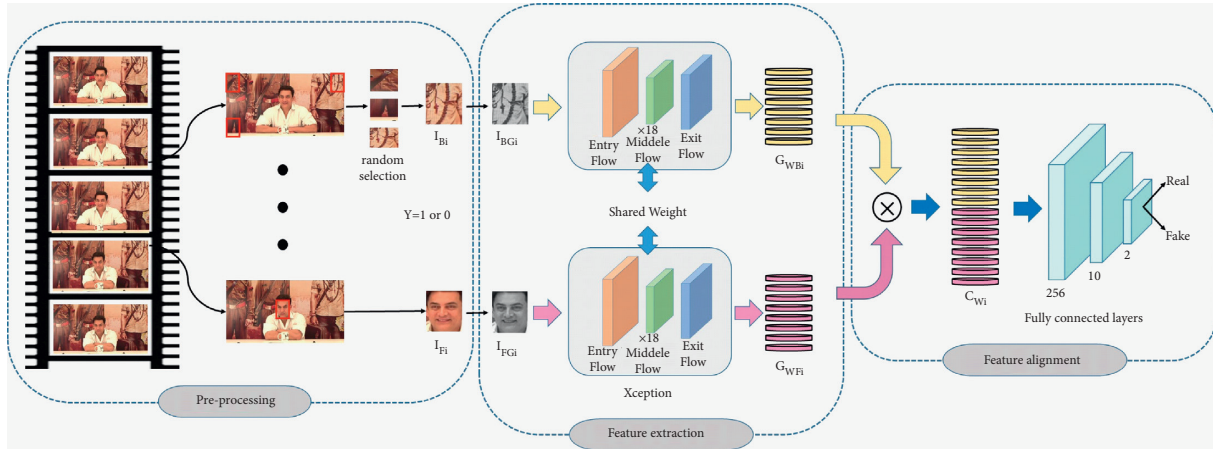


FIGURE 1: Overview of the proposed method. Our detection framework includes three modules. The preprocessing module is used to crop face area patches and background patches of video frames, where I_{Bi} and I_{Fi} ($i = 1, 2, \dots, N$) represent the face patch and the background patch, respectively. N represents the number of videos. The feature extraction module converts patch pairs into greyscale, i.e., I_{BGi} and I_{FGi} and then extracts features in the grey space of the pairs, i.e., G_{WB_i} and G_{WF_i} , using a two-stream network with shared weight. The feature alignment module mines their similarity by concatenating features from different areas and obtains the final classification result. C_{Wi} denotes aligned features. \otimes represents the concatenation operation. The final classification result is obtained after C_{Wi} goes through three fully connected layers.

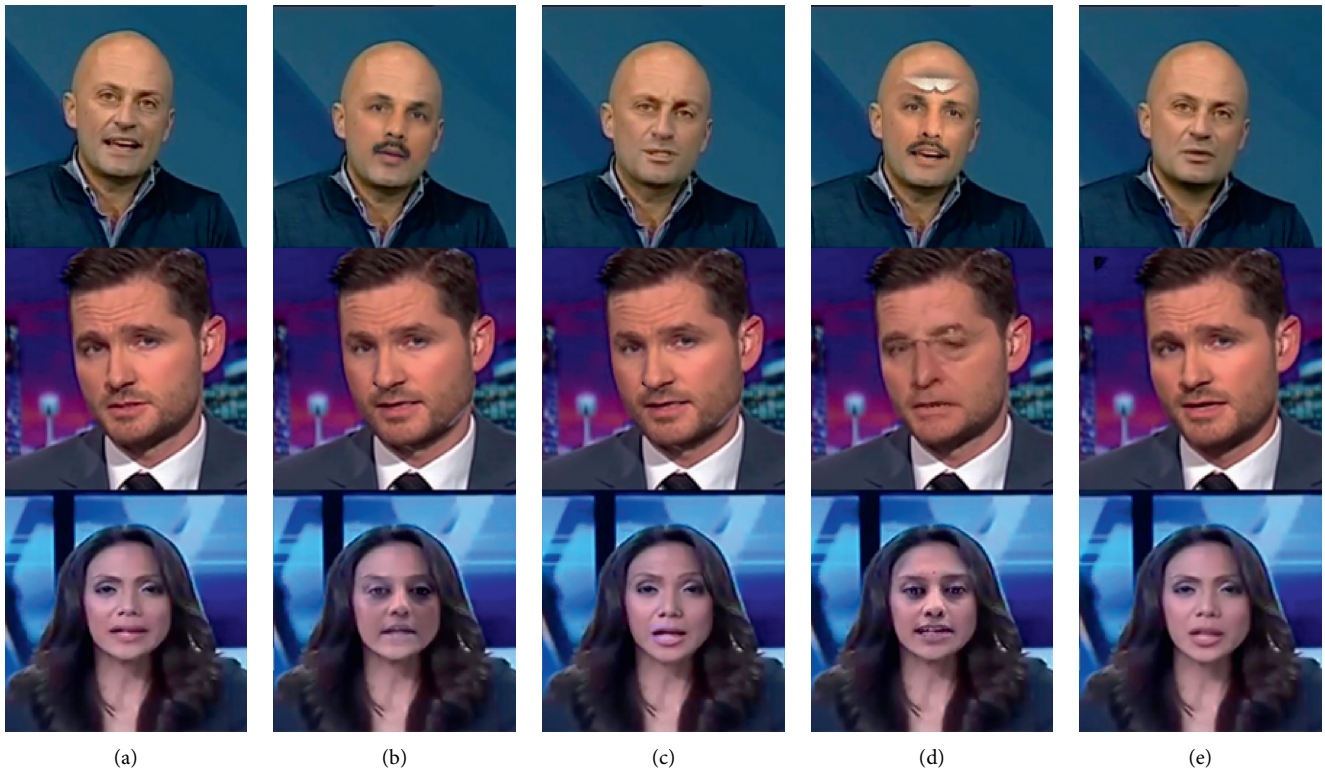


FIGURE 2: Examples of genuine images and four tampering methods. (a) Genuine images. (b) Deepfake. (c) Face2Face. (d) FaceSwap. (e) Neural Textures.

of the backbone network or attention network and good performance has been achieved. It is emphasized that methods based on features of the individual image can also be used to determine the authenticity of the videos.

2.2.2. *Detection Methods for Tampered Videos.* This type of method mainly uses the continuity and consistency of various features between video frames to determine authenticity. Therefore, it relies on the timing of the video

frames, and the detection object can only be a video, not a single image [30–38]. Haliassos et al. [38] proposed a detection called LipForensics which targets high-level semantic irregularities in mouth movements, which are common in many generated videos. But it requires a large-scale labelled dataset for pretraining. Zheng et al. [32] explored taking full advantage of the temporal coherence for video face forgery detection utilizing a novel end-to-end framework, which consists of two major stages. The temporal consistency of video frames is also used in [30–33]. Li et al. [36] proposed a long-term recurrent convolutional network (LRCN) to detect the blinking frequency of people in the videos and to compare it with the blinking frequency of normal people to distinguish between genuine and tampered videos. However, because the blinking frequency in high-quality tampered videos is almost the same as that of normal people, the prospective application of this method is not ideal. Agarwal et al. [37] used an open-source facial behavior analysis toolkit, Openface, to model the faces of five political celebrities in order to distinguish the authenticity of the videos. However, because there are not as many genuine and tampered videos for ordinary people as for politicians, this method has limited applications.

2.3. Siamese Network. The Siamese network is used to learn a function that maps the inputs into a target space such that the L_1 norm in the target space approximates the semantic distance in the input space. The details of the architecture are given in Figure 3. X_1 and X_2 are the inputs shown to the network, W is the shared parameter vector between CNNs, and $G_W(X_1)$ and $G_W(X_2)$ are the two points in the low-dimensional space that are generated by mapping X_1 and X_2 . E_W is a function that measures the compatibility between X_1 and X_2 .

3. Proposed Method

As shown in Figure 1, our method consists of three modules, i.e., preprocessing (Subsection III-A), feature extraction (Subsection III-B), and feature alignment (Subsection III-C). In addition, we introduce the voting principle in Subsection III-D.

3.1. Preprocessing. The feature extraction module takes image patch pairs as input, so we need to crop each video frame into image patch pairs. For each video in the datasets, we first use the software package dlib [47] to detect the face area of each frame in the video, and we crop a fixed-size face image patch according to the center of the face. We crop three corner background patches of the image to the same size as face image patches, excluding the lower right corner. It should be noted that the three corners are selected to facilitate cropping and improve the efficiency of preprocessing. In fact, it can be cropped anywhere on the background of images. And the number of cropped background patches can also be any odd number which is convenient for the voting principle (which will be introduced in Subsection D) other than three. The three

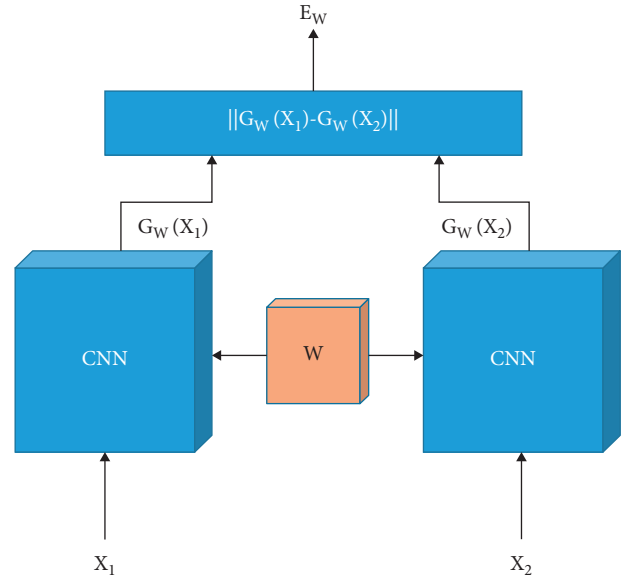


FIGURE 3: The architecture of the Siamese network.

background patches are later used by the voting principle to calibrate our test results. In the preprocessing stage, we finally process the videos in each dataset $F_i (i = 1, 2, \dots, N)$, where N represents the number of videos, into face patches I_{Fi} and background patches I_{Bi} , as described in detail in Algorithm 1.

3.2. Feature Extraction. Artefacts may be left on videos due to hardware and software differences and manufacturing imperfections. In one genuine video, the artefacts are consistent and continuous in general, such that the facial area and the background have high similarity, while in the tampered video (e.g., generated by Deepfake and FaceSwap), the similarity between the face area and the background is lower. The tampered videos generated by Face2Face and Neural Textures only modify the facial expression and some attributes and are not directly derived from different videos, but the tampering still impacts the consistency of the artefacts.

To this end, we use the improved Siamese network to measure the similarity between the face area and the background of the video frames. We employ the Xception network [48] as the backbone of the Siamese network. The Xception network is currently one of the most effective and widely used networks, as in [1, 19, 20, 22], for face forgery detection. The advantage of deep learning lies in its powerful computing ability and autonomous learning ability. Through end-to-end training and supervised learning, the convolutional neural network extracts the suitable and effective features in the grey space of the images.

After the preprocessing module, we obtain a pair of image patches. In the feature extraction module, we first convert the pair of image patches to greyscale. Since the semantic content of the face patch and the background patch is very different, greying the pair of patches can reduce the impact of the semantic content so that the network can

```

Input: Videos of the dataset:  $F_i$ ,  $i = 1, 2, \dots, N$ ,  $N$ : The number of videos
Output: Images patches  $I_B$  and  $I_F$ ,  $I_{Bi}$ : background patches,  $I_{Fi}$ : face patch
for each  $F_i$  do
  if  $F_i$  is a real video then
    Assign  $F_i$  to the set  $F_r$ 
  else
    Assign  $F_i$  to the set  $F_t$ 
    crop face patch  $I_{Fi}$  which is assigned to the set  $F_f$  and three background patches  $I_{Bi}$  which are assigned to the set  $F_b$ 

```

ALGORITHM 1: Preprocessing.

concentrate more on the low-level features with better generalization performance. Then the pair of patches are given to the Xception networks with shared weights to get the 512-dimension feature maps in the grey space. Sharing weights ensures that the two streams of the network mine the features of the same space, and at the same time it is equivalent to enriching the feature data of each stream, making the network more efficient. And the feature maps can be regarded as features of the noise distribution of the image patches.

3.3. Feature Alignment. After obtaining the features of the face patch G_{WF_i} ($i = 1, 2, \dots, N$), where N represents the number of videos, and the features of the background patch G_{WB_i} in the grey space, it is significant to measure their similarity in order to distinguish whether they are from genuine images or tampered images. The most direct way to accomplish this goal is to perform a residual operation on two feature maps, similar to what the original Siamese network does, but this is not suitable for image patches with large differences in semantic content. Thus, in the feature alignment module, we concatenate G_{WF_i} and G_{WB_i} and acquire the aligned features, which are 1024-dimensional feature maps, i.e., C_{Wi} , defined as

$$C_{Wi} = G_{WF_i} \otimes G_{WB_i}. \quad (1)$$

\otimes represents concatenating G_{WF_i} and G_{WB_i} . C_{Wi} is then input to the fully connected layers behind. There are three fully connected layers that have 256, 10, and 2 nodes in sequence. The aligned features retain all the feature information of the image patch pair so that the following fully connected layers can fully mine the similarity between them and make the learning process more stable and robust in order to achieve more satisfactory performance.

The aligned features are very robust for classification. Limited by current technical conditions, no matter what kind of face tampering technology is employed, the focus is on the continuity of semantic content, and damage to the continuity of the noise artefacts in certain feature spaces is inevitable. Therefore, compared with the genuine videos, even if tampered videos undergo a variety of postprocessing operations, the similarity between the face patch and the background patch remains at a relatively low level. Extracting the features in the grey space of the images and measuring the similarity by concatenating features greatly reduce the influence of the semantic content of the images.

This approach enables our method to maintain satisfactory detection performance for tampered images and videos with high compression factors.

Under the supervised and end-to-end training, the feature alignment module can measure the similarity between the face patch and the background patch and produce the final classification result. We train our network by minimizing the cross-entropy loss function, which is defined as

$$\text{Loss} = -[y \cdot \log(\hat{y}) + (1 - y) \cdot \log(1 - (\hat{y}))]. \quad (2)$$

y represents the labels of image patches and \hat{y} represents the classification results output by the network. The fully connected layers of the feature alignment module act as a classifier.

Algorithm 2 describes the entire training process in detail.

3.4. Voting Principle. To obtain more accurate classification results, we define a voting principle in the test stage to modify them. The difference from the training is when we randomly select an image patch, we will select three patches from the same frame as the face patch and make them form three patch pairs by copying the face image patch with the three background patches. The three pairs of patches are then input into our trained feature extraction module and feature alignment module, and three binary predicted labels are obtained. Finally, according to the voting principle that the minority obeys the majority, the predicted label, that is, the classification result of the image to which the face patch and background patches belong, is obtained. At the same time, as we emphasized in Subsection III-A, the number of background patches can also be any odd number other than three. The details of the voting principle can be found in Figure 4. Let I_F be the face patch and let I_{B1} , I_{B2} , and I_{B3} be the three corresponding background patches. Let Y_1 , Y_2 , Y_3 , and Y_t be the prediction labels of three patch pairs and the final prediction label, respectively. $Y_t = 1$ means that the image is genuine and $Y_t = 0$ means that image is tampered. Table 1 illustrates the voting principles between Y_t and the labels of the three patch pairs.

4. Experiments

In this section, we first introduce the datasets that we used in the experiment, and then, we introduce our experimental setup and detailed training process. Finally, we report the


```

Input: The pair of image patches:  $I_{Fi}$  and  $I_{Bi}$ ,  $i = 1, 2, \dots, N$ 
           $N$ : The number of videos
Output: The prediction label  $Y_{ti}$ ,  $I_{Bi}$ : background patches,  $I_{Fi}$ : face patch
while epoch  $\leq 30$  do
  for each pair of  $I_{Fi}$  and  $I_{Bi}$  do
    if  $I_{Fi}$  and  $I_{Bi}$  are from  $F_r$  then
      label  $l = 1$ 
    else
       $I_{Fi}$  and  $I_{Bi}$  are from  $F_b$ , label  $l = 0$ 
      Greying  $I_{Fi}$  to  $I_{FGi}$  and  $I_{Bi}$  to  $I_{BGi}$ 
      Mapping  $I_{FGi}$  to  $G_{W_{Fi}}$  and  $I_{BGi}$  to  $G_{W_{Bi}}$  with shared weights  $W$ 
      Concatenating  $G_{W_{Bi}}$  and  $G_{W_{Fi}}$  to  $C_{Wi}$ 
      Mapping  $C_{Wi}$  to get label  $Y_{ti}$ 
    return Siamese network model

```

ALGORITHM 2: Training.

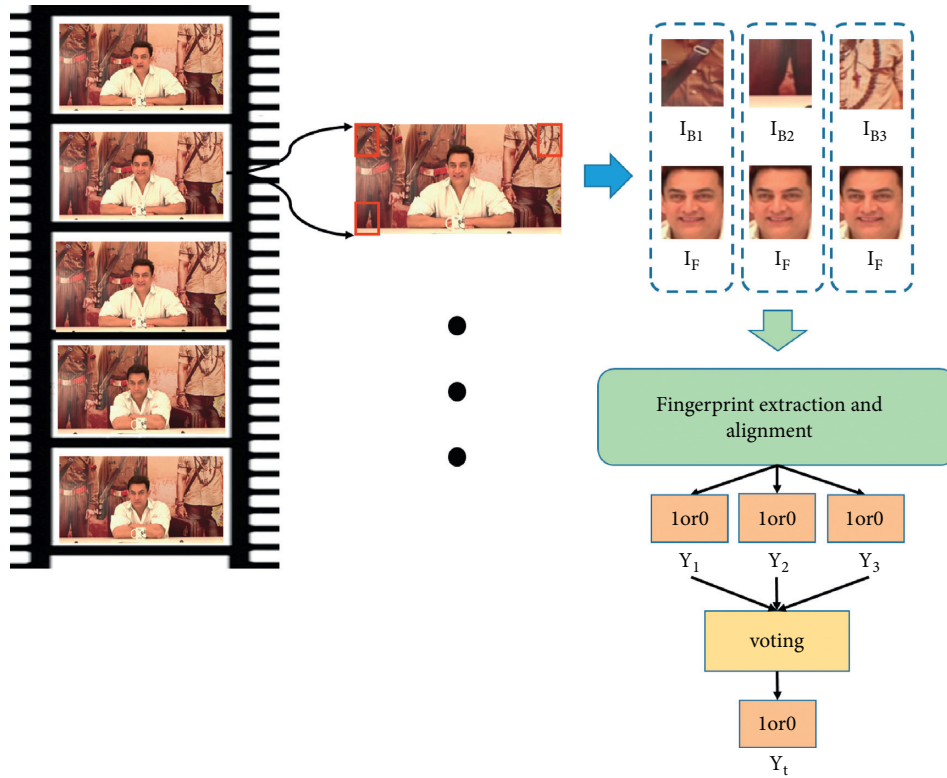


FIGURE 4: The details of the voting principle. It is used in the test stage to modify the classification results. I_F is the face patch, while I_{B1} , I_{B2} , and I_{B3} are the three corresponding background patches. Y_1 , Y_2 , and Y_3 are the prediction labels of three patch pairs, and the final predicted label, Y_t , is obtained according to the voting principle from Y_1 , Y_2 , and Y_3 .

TABLE 1: The voting principles between Y_t and Y_1 , Y_2 , and Y_3 . $Y_t = 1$ means that the image is genuine and $Y_t = 0$ means that the image is tampered.

Y	Binary label							
Y_1	1	1	1	0	1	0	0	0
Y_2	1	1	0	1	0	1	0	0
Y_3	1	0	1	1	0	0	1	0
Y_t	1	1	1	1	0	0	0	0

The bold results are the final prediction label.

performance of our proposed method and analyze the experimental results in detail.

4.1. Datasets. We used three datasets in our experiments: the FaceForensics++ dataset [1], the Celeb-DF(v2) dataset [40], and the UADFV dataset [23]. FaceForensics++ is a forensics dataset consisting of 1000 original video sequences that have been manipulated with four automated face manipulation methods: Deepfake (DP), Face2Face (F2), FaceSwap (FS),

and Neural Textures (NT); i.e., it contains four subdatasets. The data have been sourced from 977 YouTube videos, and all videos contain a trackable mostly frontal face without occlusions, which enables automated tampering methods to generate realistic forgeries. All videos have three resolutions, i.e., raw quality without compression, high quality with a light compression using a quantization of 23, and low quality with a heavy compression using a quantization of 40.

The UADFV dataset contains 98 videos, with 49 genuine videos and 49 tampered videos. All tampered videos are generated by the method of Deepfake. Each video has one subject and lasts approximately 11 seconds, with a typical resolution of 294 500 pixels. The Celeb-DF(v2) dataset is a large-scale challenging dataset for Deepfake forensics. It includes 590 original videos collected from YouTube, with subjects of different ages, ethnic groups, and genders, and 5639 high-quality Deepfake videos generated using an improved synthesis process. The overall visual quality of the synthesized Deepfake videos in the Celeb-DF dataset is greatly improved when compared to other datasets, with significantly fewer notable visual artefacts. In addition, the genuine video shows a wide range of changes in the subject’s face size, orientation, lighting conditions, and background.

4.2. Implementation Details. In our experiment, we used the software package dlib [47] to detect faces in the frames of the videos and extract the face area, but we decided to eliminate some videos in the datasets for which the face extraction failed. For every subdataset of the FaceForensics++ dataset, we select 976 tampered videos, among which 681 videos were used as the training set, 145 videos are used as the validation set, and the other 145 videos are used as the test set. For the UADFV dataset, we selected 43 tampered videos, of which 31 videos are used as the training set, 6 videos are used as the validation set, and the other 6 videos are used as the test set. The number of genuine videos is the same as the number of tampered videos. In each video, we randomly select 50 pairs of face patches and background patches for training and 150 pairs of patches for validation and testing due to the need for the voting principle.

For the Celeb-DF(v2) dataset, because the number of genuine videos is far less than that of tampered videos, we use two methods to divide the dataset in order to ensure the balance of genuine and tampered data during the training process. One method is to divide the data according to the quantity balance; that is, for both genuine and tampered videos, 400 videos are selected for training, 50 videos for validation, and 50 videos for testing, and 50 pairs of patches are randomly selected in each video for training and 150 pairs of patches are selected for validation and testing. The other method is based on the proportional balance; that is, the genuine videos are divided in the same way as the previous method, but for tampered videos, 4,000 are selected for training, 500 for validation, and 500 for testing, while only 5 pairs of patches for training and 15 pairs of patches for validation and testing in each video are randomly selected in order to keep the quantities of tampered data and genuine data the same. The precise numbers of the patch pairs for each dataset can be found in Table 2.

TABLE 2: Precise numbers of patch pairs for training, validation, and testing of the three datasets.

Dataset	Number of pairs		
	Training	Validation	Test
FaceForensics++ [1]	68600	43500	43500
UADFV [23]	3100	1800	1800
Celeb-DF(v2) [40]	40000	15000	15000

All networks have been implemented with Python 3.7 using PyTorch. Weight optimization of the network is achieved with successive batches of 16. The sizes of face patches and background patches are both 256 256. The networks are optimized via Adam [49] with default parameters ($\beta_1 = 0.9$ and $\beta_2 = 0.999$). We adjust the learning rate by combining warm-up and stepwise methods. We set the base learning rate as 0.0001. Every training process contains 30 epochs: 10 are used to warm-up, 10 are maintained at the base learning rate, and then, the learning rate is divided by 10 every 5 epochs.

4.3. Evaluation Metrics. We apply the accuracy score (Acc) and the area under the receiver operating characteristic (ROC) curve (AUC) values that are commonly used in face forgery detection as our evaluation metrics. In addition, we apply precision (P), recall (R), and the F1 score on the challenging low-quality data from the FaceForensics++ dataset [1] to better evaluate the performance of our method.

4.4. Results. We first compare the performance of our network with the three most widely used networks based on the four subdatasets of the FaceForensics++ dataset with different quality. The results are listed in Table 3.

As these results show, except for the subdataset of Neural Textures (NT) with high quality, our method outperforms all the reference methods and different face manipulation methods with respect to all quality settings. It is worth noting that our method achieves Acc values of 84.14%, 97.97%, 98.88%, and 98.21% on the subdatasets of Deepfake (DP), Face2Face (F2), FaceSwap (FS), and Neural Textures (NT) with low quality, respectively. The performance of our method far exceeds that of the reference methods; in particular, the performance becomes even better after use of the voting principle to correct the results, with values of 84.14%, 96.62%, 99.49%, and 98.90% achieved. Moreover, compared to the results on the same subdataset with raw quality and high quality, the Acc scores of reference methods have significantly declined. However, except on the DP subdataset, the performance of our method on low-quality datasets is close to that of raw quality. The previous methods for face forgery detection can mine the differences in feature distribution between genuine and tampered images to find the traces of tampered images. The image compression eliminates the forgery traces to a certain extent so that the differences in the feature distribution of genuine and tampered images are reduced. Therefore, the performance of the network will also be reduced accordingly. However, our method determines the authenticity of the images by

TABLE 3: Acc score on the FaceForensics++ dataset. LQ represents low quality, HQ represents high quality, and Raw represents raw quality.

Method	Dataset											
	LQ				HQ				Raw			
	DP	F2	FS	NT	DP	F2	FS	NT	DP	F2	FS	NT
Meso4 [11]	77.68	83.65	79.92	77.74	89.77	94.25	95.50	78.70	96.37	97.95	98.17	93.30
MesoInception4 [11]	74.20	78.75	79.72	67.94	83.74	91.48	94.34	75.06	88.34	97.65	97.81	92.52
Xception [48]	83.70	87.21	83.17	87.90	95.15	97.07	95.96	87.99	98.31	97.75	98.10	96.45
Our method	84.14	97.97	98.88	98.21	95.79	97.11	97.37	84.69	98.72	97.91	98.75	97.33
Our method (voting)	84.14	98.62	99.49	98.90	95.77	97.12	97.37	84.71	98.72	97.92	98.77	98.18

The bold results show the best.

comparing the similarity between the face area and the background of the images, which greatly enhances the robust performance of the network so that postprocessing similar to image or video compression has a relatively small impact on the performance. In addition, from the results in Table 3, it can be concluded that the voting principle does not achieve better results on the DP subdataset; specifically, on the datasets with raw and low quality, the Acc scores are equal to those of the method not employing the voting principle, and on the high-quality dataset, the score even becomes slightly lower. Overall, however, the voting principle is still beneficial to the results.

We then evaluate our method on the UADFV and Celeb-DF(v2) datasets. The results are shown in Table 4. The proposed method achieves 99.94% Acc performance on the UADFV dataset, and the score even reaches 1.00 by voting, although this is only a small improvement compared to the Xception network. With respect to the ways that the Celeb-DF(v2) dataset is divided according to the proportional balance and the quantity balance, our method achieves 92.61% and 94.94%, respectively, exhibiting remarkable improvement compared to the reference methods. These results prove the superiority of our method.

To better evaluate the performance of our method on the low-quality datasets, we calculate precision (P), recall (R), and the F1 score of all methods, as shown in Table 5, and generate ROC curves of different methods as shown in Figure 5 on the FaceForensics++ dataset with low quality. It can be seen from the results in Table 5 that, compared with the reference methods, our method has achieved better performance with respect to all evaluation metrics on the four subdatasets. The AUC values of the proposed method, i.e., the area values in Figure 5, are far ahead, with the exception that the results on the DP subdataset are close to those of Xception.

4.4.1. Comparison with Recent Works on the Low-Quality Datasets of FaceForensics++ [1]. In order to demonstrate the competitive results of our method on low-quality datasets, we compared our results with recent methods [14, 19, 20, 22, 25, 28, 39, 50–52]. Since the experimental sets between us and others are almost the same, we directly used the results in these papers. The results are shown in Table 6.

Accuracy scores marked in bold represent the highest accuracy scores. The Acc of our method in some categories exceeds all the reference methods, i.e., F2, FS, NT. These results fully demonstrate that our method exhibits very good

TABLE 4: Acc score on the UADFV and Celeb-DF(v2) datasets. C P and C Q represent the way in which the Celeb-DF dataset is divided according to the proportional balance and the quantity balance, respectively.

Method	Dataset		
	UADFV [23]	Celeb-DF(v2) [40]	
		C _P	C _Q
Meso4 [11]	82.67	87.10	83.75
MesoInception4 [11]	96.33	88.10	70.15
Xception [48]	99.33	90.78	89.64
Our method	99.94	92.61	94.94
Our method (voting)	100.00	92.62	94.93

and robust performance and generalization ability on challenging low-quality datasets and that the impact of compression processing is very small, which is extremely important for the practical application and promotion of the detection methods.

4.5. Discussion on Other Influencing Factors

4.5.1. Effect of Size of Image Patches. To evaluate the impact of image patch size on network performance, we used the patch sizes of 256×256 , 192×192 , and 128×128 to conduct ablation tests on the FaceForensics++ dataset with low quality. The results are shown in Table 7. The size of the image patches exerts an obvious influence on the performance of our method. For the size of 256×256 , our method including the voting principle achieves the leading performance on all datasets, but for the sizes of 192×192 and 128×128 , our method offers better performance on only three datasets. The impact of the size also differs according to different tampering methods. For Deepfake, the result for the size of 192×192 is the best, but for the other three methods, the results are best for the size of 256×256 . In general, our method performs best for the size of 256×256 .

4.5.2. Effects of Different Tampering Methods. From Table 3, it can be concluded that our method has a higher accuracy rate for the tampered images generated by FaceSwap with different quality. This is because FaceSwap has a simpler production principle and process than the other three methods. The most difficult tampering methods to detect for our method are Deepfake, Neural Textures, and Face2Face on the datasets of

TABLE 5: Precision (P), Recall (R), and $F1$ score on the FaceForensics++ dataset with low quality.

Method	P				R				$F1$			
	DP	$F2$	FS	NT	DP	$F2$	FS	NT	DP	$F2$	FS	NT
Meso4 [11]	77.88	82.71	78.07	80.13	77.32	85.08	83.21	73.77	77.60	83.88	80.56	76.82
MesoInception4 [11]	80.91	87.91	92.49	86.48	63.35	66.68	64.69	42.54	71.06	75.83	76.13	57.03
Xception [48]	82.96	86.01	81.41	85.02	84.82	88.88	85.97	92.01	83.88	87.42	83.63	88.38
Our method	83.36	98.15	98.85	98.34	85.84	97.78	98.91	97.91	84.58	97.97	98.88	98.12

The bold results show the best.

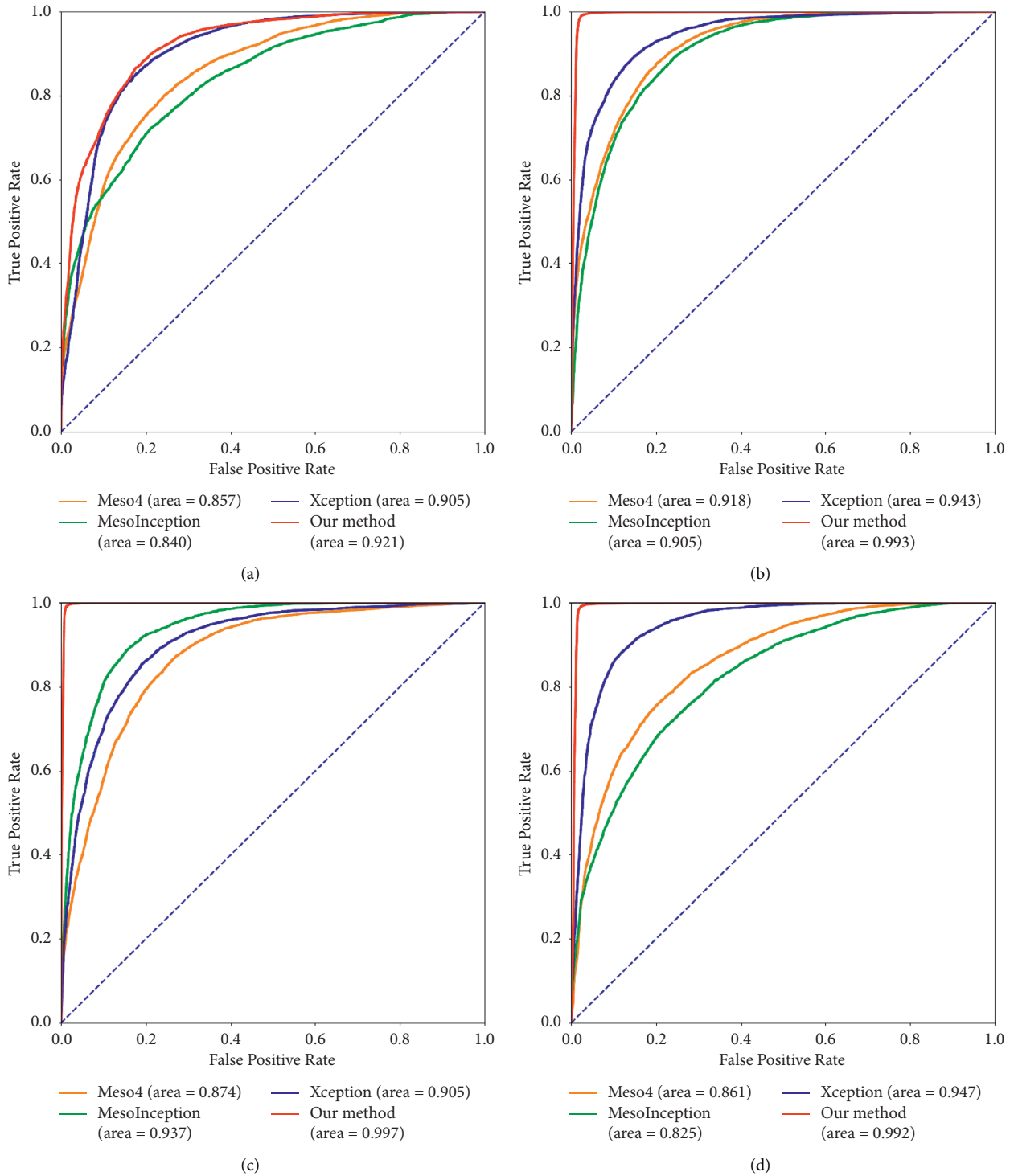


FIGURE 5: ROC curves of different methods based on the FaceForensics++ dataset with low quality. (a) Deepfake. (b) Face2Face. (c) FaceSwap. (d) Neural Textures.

TABLE 6: Comparative analysis of detection performance with recent methods on the low-quality datasets of FaceForensics++ [1]. The performances of [19, 25, 28, 39], [50, 51, 52] are obtained from [28], and others are from the original papers, respectively.

Method	Dataset			
	DP	F2	FS	NT
Durall et al. [50]	71.69	65.66	65.43	59.34
DSP-FWA [25]	93.60	91.77	90.73	83.15
Liu et al. [51]	92.39	90.67	91.99	84.69
Qian et al. [19]	96.01	93.62	94.33	86.37
Bondi et al. [52]	94.95	91.33	94.26	87.79
Bonettini et al. [39]	96.13	92.93	94.09	88.15
Khalid et al. [14]	88.40	71.20	86.10	97.50
Liu et al. [22]	93.48	86.02	92.26	76.78
Luo et al. [20]	98.60	95.70	92.90	—
Yang et al. [28]	97.88	96.85	96.87	88.47
Our method	84.14	97.97	98.88	98.21
Our method (voting)	84.14	98.62	99.49	98.90

The bold results show the best.

TABLE 7: Acc scores of different sizes of image patches based on the FaceForensics++ dataset with low quality.

Method	Dataset											
	256 × 256				192 × 192				128 × 128			
	DP	F2	FS	NT	DP	F2	FS	NT	DP	F2	FS	NT
Meso4 [11]	77.68	83.65	79.92	77.74	56.16	55.54	61.98	51.81	57.59	56.64	56.26	50.06
MesoInception4 [11]	74.20	78.75	79.72	67.94	76.23	64.25	63.46	71.40	67.97	64.15	70.22	64.41
Xception [48]	83.70	87.21	83.17	87.90	78.47	67.84	71.95	82.86	77.26	61.75	73.86	63.59
Our method	84.14	97.97	98.88	98.21	84.74	66.33	78.52	94.01	76.05	65.18	74.06	79.31
Our method (voting)	84.14	98.62	99.49	98.90	84.74	66.34	78.54	95.74	76.06	65.17	74.15	80.72

The bold results show the best.

low quality, high quality, and raw quality, respectively. Therefore, different tampering methods should be tested with different preprocessing operations in practical applications.

4.5.3. Effects of Different Image Modes. In our basic experiment, we have processed all image patches into greyscale mode. To compare the impacts of different image modes on the classification performance, we used the image patches of the RGB mode to conduct a comparative experiment. The experiment is performed on the FaceForensics++ dataset with low quality. Figure 6 shows the results of the comparison experiment. It can be determined that the classification performance in the greyscale mode is better than that in the RGB mode for each subdataset, and it is even more superior than Face2Face and FaceSwap. This result shows that our method can find a more suitable feature distribution in the grey space to distinguish between real and tampered images. And the reason may be that the grey domain reduces the relevant semantic features produced by colors compared to the RGB domain, so that our network can find more general features.

4.5.4. Effect of Concatenating Features. We use feature subtraction instead of concatenation in the feature alignment module to conduct a comparative experiment, and the experiment is performed on the FaceForensics++ dataset with low quality. The results are shown in Figure 7. It is

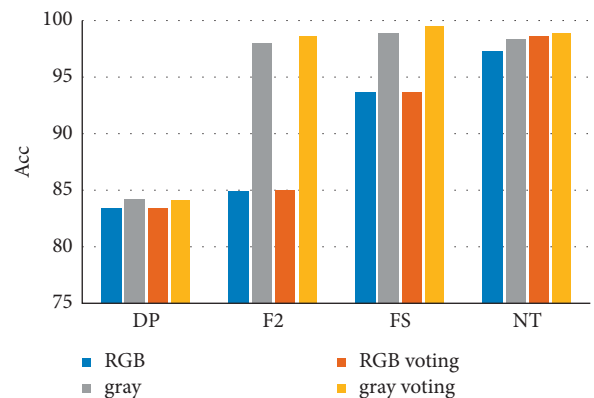


FIGURE 6: Results for the impacts of different image modes on classification performance. The classification performance in the greyscale mode is better than that in the RGB mode on all sub-datasets, especially for Face2Face and FaceSwap.

obvious that concatenating features is more effective. In fact, the subtraction operation is more suitable for use in face recognition tasks with image pairs including similar semantic content. In our task, the face area and the background area are divergent in semantic content, the effect of the subtraction operation is greatly reduced, and the effect is almost completely lost for Face2Face and FaceSwap. The concatenation operation allows the fully connected layer to be classified under richer feature conditions, resulting in better performance.

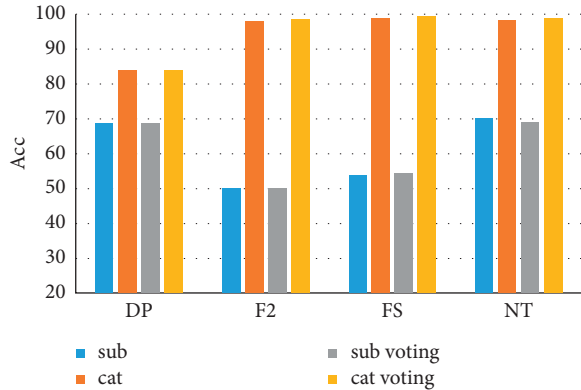


FIGURE 7: The results of the impact of concatenating features on classification performance. In the figure, cat represents concatenation and sub represents subtraction. The classification performance of the method with feature concatenation far exceeds that of the method with features subtraction on all subdatasets.

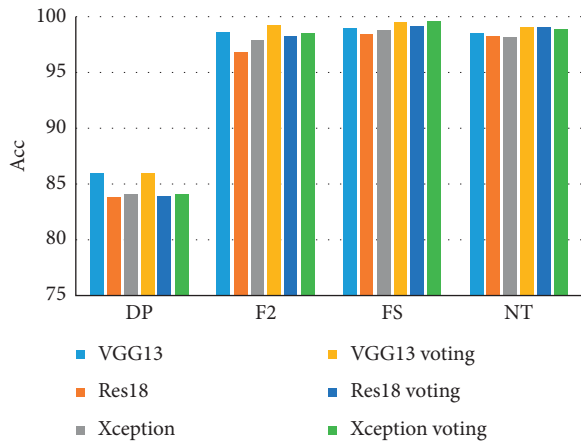


FIGURE 8: The results of the impacts of different backbones of the feature extraction module on classification performance. The classification performance of the framework used in VGG13 has achieved better results. These results illustrate the universality of our method for different classification networks.

4.5.5. Effects of Different Backbones of the Feature Extraction Module. We chose Xception [48], which is currently the most widely used network in the field of face forgery detection, as the backbone of the feature extraction module. However, the backbone of our feature extraction module based on the Siamese framework can also be some general classification networks. To explore the universality of our method, we use VGG13 [53] and ResNet18 [54] to conduct a comparative experiment: the experiment is performed on the FaceForensics++ dataset with low quality. As shown in Figure 8, the overall performance of the detection framework using Xception because of the backbone is slightly better than that of ResNet18, but slightly worse than VGG13. This finding shows to a certain extent that our method still has the potential to continue to improve and that it can be adapted to some general classification networks.

Through these ablation experiments, we explore the impacts of different conditions on our methods. At the same

time, it can also be learned that, for images and videos with different resolution and those generated by different forgery methods, we should use the framework with different details to achieve the best results. The generalization performance of the method will be the focus of future work.

5. Conclusion

The development of deep learning has significantly improved the quality and efficiency of generating forged face images and videos. In this paper, we propose an innovative face forgery detection framework based on the improved Siamese network, which extracts and aligns the features of the face area and the background of the image and then mines the similarity between them to determine the authenticity of the image. This framework not only offers great robustness and generalization performance but also makes full use of the feature information of the image background. We evaluate our method on several different datasets, thus proving its effectiveness in practice, especially that it achieves impressive results on low-quality datasets.

Data Availability

The data used to support the findings of the study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no financial and personal relationships with other people or organizations that can inappropriately influence their work; there is no professional or other personal interest of any nature or kind in any product, service, and/or company that could be construed as influencing the position presented in, or the review of, the manuscript entitled.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (nos. U1936117, 62106037, 62076052, and 61772111), the Science and Technology Innovation Foundation of Dalian (no. 2021JJ12GX018), the Open Project Program of the National Laboratory of Pattern Recognition (NLPR) (no. 202100032), and the Fundamental Research Funds for the Central Universities (DUT21GF303, DUT20TD110, and DUT20RC(3)088).

References

- [1] A. Roßler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, "Faceforensics++: learning to detect manipulated facial images," in *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1–11, Seoul, South Korea, January 2019.
- [2] Deepfakes, "Deepfakes," 2018, <https://github.com/deepfakes/>.
- [3] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Nießner, "Face2face: real-time face capture and reenactment of rgb videos," *Communications of the ACM*, vol. 62, pp. 96–104, 2019.
- [4] Faceswap, "Faceswap," 2018, <https://github.com/>.

- [5] J. Thies, M. Zollhofer, and M. Nießner, "Deferred neural rendering: image synthesis using neural textures," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 4, pp. 1–12, 2019.
- [6] B. Chen, W. Tan, G. Coatrieux, Y. Zheng, and Y.-Q. Shi, "A serial image copy-move forgery localization scheme with source/target distinguishment," *IEEE Transactions on Multimedia*, vol. 23, pp. 3506–3517, 2021.
- [7] Y. Zhang, L. Zheng, and V. L. L. Thing, "Automated face swapping and its detection," in *Proceedings of the 2017 IEEE 2nd International Conference on Signal and Image Processing (ICSIP)*, pp. 15–19, Singapore, November 2017.
- [8] M. Koopman, A. Macarulla Rodriguez, and Z. Geradts, "Detection of deepfake video manipulation," in *Proceedings of the 20th Irish Machine Vision and Image Processing Conference (IMVIP)*, pp. 133–136, Belfast, Ireland, 2018.
- [9] A. Khodabakhsh, R. Raghavendra, K. Raja, P. Wasnik, and C. Busch, "Fake face detection methods: can they be generalized?" in *Proceedings of the 2018 International Conference of the Biometrics Special Interest Group (BIOSIG)*, pp. 1–6, Germany, September 2018.
- [10] P. Korshunov and S. Marcel, "Deepfakes: a new threat to face recognition? assessment and detection," 2018, <https://arxiv.org/abs/1812.08685>.
- [11] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, "Mesonet: a compact facial video forgery detection network," in *Proceedings of the 2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, pp. 1–7, Hong-Kong, China, December 2018.
- [12] H. H. Nguyen, F. Fang, J. Yamagishi, and I. Echizen, "Multi-task learning for detecting and segmenting manipulated facial images and videos," in *Proceedings of the 2019 IEEE 10th International Conference on Biometrics Theory, Applications and Systems (BTAS)*, pp. 1–8, Tampa, FL, USA, 2019.
- [13] H. H. Nguyen, J. Yamagishi, and I. Echizen, "Capsule-forensics: using capsule networks to detect forged images and videos," in *Proceedings of the ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2307–2311, Brighton, UK, May 2019.
- [14] H. Khalid and S. S. Woo, "Oc-fakedect: classifying deepfakes using one-class variational autoencoder," in *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 2794–2803, June 2020.
- [15] P. Zhou, X. Han, V. Morariu, and L. Davis, "Two-stream neural networks for tampered face detection," in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 1831–1839, Honolulu, HI, USA, July 2017.
- [16] H. Zhao, W. Zhou, D. Chen, T. Wei, W. Zhang, and N. Yu, "Multi-attentional deepfake detection," in *Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2185–2194, 2021.
- [17] C. Wang and W. Deng, "Representative forgery mining for fake face detection," in *Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14 918–1014 927, Nashville, TN, USA, June 2021.
- [18] B. Chen, X. Liu, Y. Zheng, G. Zhao, and Y.-Q. Shi, "A robust gan-generated face detection method based on dual-color spaces and an improved xception," in *Proceedings of the IEEE Transactions on Circuits and Systems for Video Technology*, November 2021.
- [19] Y. Qian, G. Yin, L. Sheng, Z. Chen, and J. Shao, "Thinking in frequency: face forgery detection by mining frequency-aware clues," in *Proceedings of the European Conference on Computer Vision*, pp. 86–103, Springer, Glasgow, UK, 2020.
- [20] Y. Luo, Y. Zhang, J. Yan, and W. Liu, "Generalizing face forgery detection with high-frequency features," in *Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16 312–316 321, Nashville, TN, USA, June 2021.
- [21] J. Li, H. Xie, J. Li, Z. Wang, and Y. Zhang, "Frequency-aware discriminative feature learning supervised by single-center loss for face forgery detection," in *Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6454–6463, 2021.
- [22] H. Liu, X. Li, W. Zhou et al., "Spatial-phase shallow learning: rethinking face forgery detection in frequency domain," in *Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 772–781, 2021.
- [23] X. Yang, Y. Li, and S. Lyu, "Exposing deep fakes using inconsistent head poses," in *Proceedings of the ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 8261–8265, Brighton, UK, May 2019.
- [24] F. Matern, C. Riess, and M. Stamminger, "Exploiting visual artifacts to expose deepfakes and face manipulations," in *Proceedings of the 2019 IEEE Winter Applications of Computer Vision Workshops (WACVW)*, pp. 83–92, Waikoloa Village, HI, USA, February 2019.
- [25] Y. Li and S. Lyu, "Exposing deepfake videos by detecting face warping artifacts," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 46–52, IEEE, Long Beach, CA, USA, June 2019.
- [26] X. Zhu, H. Wang, H. Fei, Z. Lei, and S. Li, "Face forgery detection by 3d decomposition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 46–52, Nashville, TN, USA, June 2021.
- [27] L. Li, J. Bao, T. Zhang et al., "Face x-ray for more general face forgery detection," in *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5000–5009, Seattle, WA, USA, March 2020.
- [28] J. Yang, A. Li, S. Xiao, W. Lu, and X. Gao, "Mtd-net: learning to detect deepfakes images by multi-scale texture difference," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 4234–4245, 2021.
- [29] J. Yang, S. Xiao, A. Li, G. Lan, and H. Wang, "Detecting fake images by identifying potential texture difference," *Future Generation Computer Systems*, vol. 125, pp. 127–135, 2021.
- [30] D. Guera and E. Delp, "Deepfake video detection using recurrent neural networks," in *Proceedings of the 2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pp. 1–6, February 2018.
- [31] E. Sabir, J. Cheng, A. Jaiswal, W. AbdAlmageed, I. Masi, and P. Natarajan, "Recurrent convolutional strategies for face manipulation detection in videos," in *Proceedings of the CVPR Workshops*, Long Beach, CA, USA, June 2019, <https://arxiv.org/abs/1905.00582>.
- [32] G. Jia, M. Zheng, C. Hu et al., "Inconsistency-aware wavelet dual-branch network for face forgery detection," *IEEE Transactions on Biometrics, Behavior, and Identity Science*, vol. 3, no. 3, pp. 308–319, 2021.
- [33] Y. Zheng, J. Bao, D. Chen, M. Zeng, and F. Wen, "Exploring temporal coherence for more general video face forgery detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15 044–115 054, Montreal, Canada, October 2021.

- [34] I. Amerini, L. Galteri, R. Caldelli, and A. D. Bimbo, "Deepfake video detection through optical flow based cnn," in *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pp. 1205–1207, Seoul, South Korea, March 2019.
- [35] U. A. Ciftci and I. Demir, "Fakecatcher: detection of synthetic portrait videos using biological signals," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 99, 2020.
- [36] Y. Li, M. C. Chang, and S. Lyu, "In ictu oculi: exposing ai created fake videos by detecting eye blinking," in *Proceedings of the 2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, pp. 1–7, Hong Kong, China, December 2018.
- [37] S. Agarwal, H. Farid, Y. Gu, M. He, K. Nagano, and H. Li, "Protecting world leaders against deep fakes," in *CVPR Workshops*, 2019.
- [38] A. Haliassos, K. Vougioukas, S. Petridis, and M. Pantic, "Lips don't lie: a generalisable and robust approach to face forgery detection," in *Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5037–5047, <http://arxiv.org/abs/2012.07657>, Nashville, TN, USA, June 2021.
- [39] N. Bonettini, E. D. Cannas, S. Mandelli, L. Bondi, P. Bestagini, and S. Tubaro, "Video face manipulation detection through ensemble of cnns," in *Proceedings of the 25th International Conference on Pattern Recognition (ICPR)*, pp. 5012–5019, <https://arxiv.org/abs/2004.07676>, Milano, Italy, January 2021.
- [40] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, "Celeb-df: a large-scale challenging dataset for deepfake forensics," in *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3204–3213, 2020.
- [41] B. Dolhansky, J. Bitton, B. Pflaum et al., "The deepfake detection challenge dataset," 2020, <https://www.arxiv-vanity.com/papers/2006.07397/>.
- [42] L. Jiang, W. Wu, R. Li, C. Qian, and C. C. Loy, "Deepforensics-1.0: a large-scale dataset for real-world face forgery detection," in *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2886–2895, Seattle, WA, USA, June 2020.
- [43] B. Zi, M. Chang, J. Chen, X. Ma, and Y. G. Jiang, "Wild-deepfake: a challenging real-world dataset for deepfake detection," in *Proceedings of the 28th ACM International Conference on Multimedia*, New York; NY, 2020.
- [44] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, pp. 539–546, San Diego, CA, USA, June 2005.
- [45] H. Farid, "Image forgery detection – a survey," *IEEE Signal Processing Magazine*, vol. 26, no. 2, pp. 16–25, 2009.
- [46] P. Pe' rez, M. Gangnet, and A. Blake, "Poisson image editing," *ACM Transactions on Graphics*, vol. 22, pp. 313–318, 2003.
- [47] D. King, "Dlib-ml: a machine learning toolkit," *Journal of Machine Learning Research*, vol. 10, pp. 1755–1758, 2009.
- [48] F. Chollet, "Xception: deep learning with depthwise separable convolutions," in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1800–1807, <https://arxiv.org/abs/1610.02357>, Honolulu, HI, USA, July 2017.
- [49] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," 2015, <https://arxiv.org/abs/1412.6980>.
- [50] R. Durall, M. Keuper, F.-J. Pfreundt, and J. Keuper, "Unmasking deepfakes with simple features," 2019, <https://arxiv.org/abs/1911.00686>.
- [51] Z. Liu, X. Qi, J. Jia, and P. H. S. Torr, "Global texture enhancement for fake face detection in the wild," in *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8057–8066, <https://arxiv.org/abs/2002.0013>, Seattle, WA, USA, June 2020.
- [52] L. Bondi, E. D. Cannas, P. Bestagini, and S. Tubaro, "Training strategies and data augmentations in cnn-based deepfake video detection," in *Proceedings of the 2020 IEEE International Workshop on Information Forensics and Security (WIFS)*, pp. 1–6, <https://arxiv.org/abs/2011.07792>, New York City, NY, USA, December 2020.
- [53] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015, <https://arxiv.org/abs/1409.1556>.
- [54] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, <https://arxiv.org/abs/1512.03385>, Las Vegas, NV, USA, June 2016.

Research Article

Perceptual Image Hashing Based on Multitask Neural Network

Cheng Xiong , Enli Liu, Xinran Li , Heng Yao , Lei Zhang, and Chuan Qin 

School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China

Correspondence should be addressed to Chuan Qin; qin@usst.edu.cn

Received 2 November 2021; Accepted 2 December 2021; Published 18 December 2021

Academic Editor: Beijing Chen

Copyright © 2021 Cheng Xiong et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

With the advent of the era of multimedia and in-depth development, the whole human society has been produced and spread a huge amount of image data, but at the same time, in view of the digital image and tamper with the attack of piracy phenomenon also more and more serious, malicious attacks will produce serious social, military, and political influence, therefore, to protect the authenticity of the original image content, which is also more and more important. In order to further improve the performance of image hashing and enhance the protection of image data, we proposed an end-to-end dual-branch multitask neural network based on VGG-19 to produce a perceptual hash sequence and used prepart of network of pretrained VGG-19 model to extract image features, and then, the image features are transformed into a hash sequence through a convolutional and fully connected network. At the same time, in order to enhance the function of the network and improve the adaptability of the proposed network to using scenarios, the rest part of the network layer of the VGG-19 model was used as another branch for image classification, so as to realize the multitask characteristics of the network. Through the experiment of the testing set, the network can not only resist many kinds of attack operations (content retention operations), but also realize accurate classification about the image, and has a satisfactory tampering detection ability.

1. Introduction

Since the beginning of this century, the technologies of Internet and multimedia develop rapidly, and information interaction mode of people has been transformed from text message to multidirectional fusion presentation of text message, image, and video information. With the wide application of powerful image editing tools, massive digital images are easily tampered; thus, the protection of the real content of the image is increasingly important. Due to the influence of technology, equipment, time, and other factors, an image is often distributed without any protection after it is produced, which makes the image more vulnerable to piracy, tamper, and other attack operations. In order to deal with a variety of malicious attacks, image hashing technology can be used to generate a unique and unidirectional perceptual image hash sequence for the original image. Image hashing, also known as “image fingerprint,” verifies image piracy or tamper by comparing the similarity of hash sequences. Imagine a scenario that there is an image that is

so important for the owner. But he does not know if his image was tampered or pirated, and it is so difficult for him to check in the image dataset, which has so many images. So, the image hashing can help him to find the similar images quickly by comparing the similarity of hash sequences effectively and judge whether they are tampered or piracy images to protect the copyright.

With the in-depth research and development of image hashing schemes, a variety of schemes have been proposed by researchers according to various requirements of multimedia security. Generally speaking, a typical perceptual image hashing scheme has the following three properties: (1) *perceptual robustness*—the generated hash sequence of the image without changing the visual content after the content retention operations should be similar or the same as the hash sequence of the original image; (2) *discriminative/anticollision*—two hash sequences correspond to two completely different images should be completely different; and (3) *security*—the hash sequence needs to have key dependence to ensure the security of scheme; that is, the hash

sequence generated with the wrong key is completely different from the sequence generated with the right key. The perceptual image hashing scheme includes three main stages: preprocessing, feature extraction, and hash generation. According to the methods of feature extraction, perceptual image hashing schemes can be divided into four main types: methods based on spatial domain, transform domain, dimensionality reduction, and deep learning.

In the methods based on the spatial domain, Schneider and Chang [1] used the histogram features of the image to generate hash sequence, which opened up the research in the field of image hashing. Yan et al. [2] proposed adaptive local feature extraction technology to obtain the location information of features and achieve image tampering positioning. In order to resist the attack of image rotation from any Angle, some scholars [3] used MDS (Multidimensional Scaling Technology) [4] on the basis of ring-based coding scheme, ring division, and invariant vector distance [5], and experimental results show that the hash sequences generated by this scheme are robust and unique for common image content retention operations. The scheme in [6] used the relationship between local feature points to overcome the problem, which feature point distribution is ignored. Qin et al. [7] proposed a robust image hashing scheme based on perceived texture and structural features; furthermore, local texture features and color vector angle features were considered simultaneously in [8]. Shen et al. [9] extracted the color opposition component from the secondary image and applied quadtree decomposition to connect the generated color feature vector with the structural feature vector and then combined with pseudo-random key scrambling to generate the final hash sequence. For the content of image in screen, the scheme in [10] extracted the maximum gradient and the corresponding direction information from R , G , and B color components and counted the relevant data to construct the image hash sequence. There are also some research studies about retrieval, and the scheme in [11] proposed a content-based image retrieval scheme in multi-user scenarios, which used Euclidean distance comparison technology to sort the similarity of image feature vectors and return top- K results to achieve the retrieval purpose. Li et al. [12] proposed an encrypted image retrieval system supporting multiple keys with edge computing based on local sensitive hashing, secure neighbor, and proxy re-encryption technologies, which improved the efficiency and accuracy of image retrieval.

In the methods based on the transform domain, some early schemes used DCT [13] (discrete cosine transform) and DFT (discrete Fourier transform) to design schemes. From the perspective of human visual characteristics, Watson was used to adjust the corresponding frequency domain coefficients to improve the robustness and discrimination of the scheme in [14]. Some scholars also used the DFT [15] to extract robust frequency features in the secondary image, and nonuniform sampling was used to combine with low and intermediate frequency components to obtain a secure hash sequence. In the work of [16], the CSLBP (center symmetric local binary pattern) was applied to DWT (discrete wavelet transform) to generate compact

image hashing. A geometric invariant vector distance method based on both the spatial domain and the frequency domain was proposed in [17]. In the dimensional-reduction method, NMF (non-negative matrix factorization), PCA (principal component analysis), and SVD (singular value decomposition) often occur as important steps. In [18], hash sequences were generated by combining BTC (block truncation coding) with PCA. A scheme based on low-rank sparse decomposition was proposed in [19].

In recent years, with the continuous improvement of GPU performance, and continuous development of deep learning, so many researchers have used deep neural networks to achieve image retrieval and the performance is much better than the traditional image retrieval schemes [20–23]. But, generally speaking, there are relatively few research studies on perceptual robust hashing used deep neural networks. In the methods based on deep neural networks, the scheme in [24] proposed a robust image hashing scheme, which is based on deep learning. And this work was an early scheme to perceptual image hashing with deep neural networks, and it has better performance than traditional schemes. In the work of [24], researchers have used pretrained DAE (auto encoder denoising) to enhance robustness and used fine-tuning to improve the accuracy of image detection. In [25], an image hashing scheme based on CNN (convolutional neural network) with multiple constraints was proposed, and the experimental results showed that it can obtain a good balance between robustness and discrimination. In order to strengthen the functional properties of the neural network and make more efficient and reasonable use of existing computing resources, there are also some other schemes about deep learning in the field of image authentication. The scheme in [26] introduced two subnetworks to improve the BusterNet for image copy-move forgery localization with source/target region distinguishment, and these subnetworks were the copy-move similarity detection network (CMSDNet) and the source/target region distinguishment network (STRDNet). The face detection is also important. Reference [27] used RGB and YCbCr color spaces, and introduced the convolutional block attention module and multilayer feature aggregation module into the Xception model to achieve better performance for detecting postprocessed face images.

We proposed a perceptual image hashing scheme based on the multitask neural network. The mainly innovation and contribution are as follows:

- (1) Efficient end-to-end framework. Instead of the traditional strong explanatory method with low efficiency and weak generalization ability, an advanced and efficient deep learning method is adopted to collect image features and generate hash sequence. Based on the excellent performance of the convolutional network and fully connected network, the end-to-end hash sequence generated framework is realized by integrating feature extractor and hash generator.
- (2) Multitask intensifies the applicability of network model to multiple scenarios. In order to improve the

applicability of the neural network model to multiple scenarios and use an excellent pretrained model, based on the pretrained model of VGG-19 neural network, we added a dual-branch layer after the feature extractor, so as to achieve the purpose of multitask.

In Section 2, the structure of the proposed network, loss function, and other related contents will be introduced. In Section 3, the experimental results of the proposed neural network based on a specific training strategy will be introduced. The advantages of our scheme will be summarized in Section 4.

2. Proposed Scheme

At present, the field of deep learning is developing rapidly, a large number of network structures have emerged, and the use of pretrained models is becoming more and more systematic. Many network structures are used for different tasks. On the basis of realizing the image hash process, the proposed scheme has added the function of image classification, so that the final neural network not only can generate the hash sequence of the image but also has function in the image classification, which is adapted to more application scenarios.

2.1. Network Structure. In the structure of the proposed neural network, we comprehensively considered the task requirements on the number of network layers and network structure, and selected VGG-19 as the basic structure to use. We first use part of VGG-19 network layer as the image feature extractor. Note that, because the function of image classification should be considered, this part of VGG-19 does not participate in the parameter updating of the training in proposed neural network, but uses the fixed parameters. The main structure of the proposed neural network is described in Figure 1. The feature extractor is connected with a small convolutional network of six layers and constructed hash generation network, which is named Branch I, and the connected point of this generated network is named branch point. The proposed network has selected the output in the second pooling of the VGG-19 network as the branch point through testing in the network training process. Specific test and evaluation will be discussed in Subsection 2.2. At the same time, the rest part (Branch II) of the network layer in VGG-19 also is remained and connected with the feature extractor through branch point to realize the function of image classification. Branch II was composed of twelve convolutional layers and three FC (Fully connected) layers, and the activation function was ReLU. So that, due to branch point, multitask is achieved and constructed the dual-branch neural network.

During the processing stage of the image hashing neural network, the feature extractor is used to collect features of the image. Then, the features are input into the small convolutional network to generate the hash sequence, and the small convolutional network is mainly composed of four blocks (convolutional layer + BN + ReLU) and two FC layers. Image features and intermediate features are so important,

and the convolutional layer is a useful filter to extract them. Thus, we used four convolutional blocks in the hashing neural network to further extract useful features, and two FC layers were used to compress and transform the features into a hash sequence. As for the other branch, it is used to process features, which can classify images and realize the multitask capability. Due to the use of FC and average pool layers, the input size of image was limited, and the size was 128×128 .

2.2. Loss Function. In order to measure the similarity of generated hash sequences, we use MSE (mean square error) as the measurement tool to calculate the hash distance among original image, similar images, and different images. However, due to the large numerical span of the generated hash sequence, direct use of MSE will make the network convergence unstable, so the activation function named Sigmoid is used to normalize the values obtained by MSE.

We define that there are n similar images and n different images (the specific composition of the dataset will be introduced in Subsection 2.3, Subsection 3.1, and Subsection 3.2), so that, in the situation of including the original image, $2n + 1$ hash sequences are generated, which is given in the following equation:

$$H(\chi), H(\hat{\chi}_{[1]}), \dots, H(\hat{\chi}_{[n]}), H(\kappa_{[1]}), \dots, H(\kappa_{[n]}), \quad (1)$$

where $H(\cdot)$ represents the whole end-to-end hashing neural network, χ is the original image, $\hat{\chi}_{[i]}$ is similar image ($i = 1, 2, \dots, n$), and $\kappa_{[i]}$ represents the different image ($i = 1, 2, \dots, n$). The hash sequences of images are generated, and the MSE is used to measure the distance of sequences, which are given as

$$\begin{aligned} \varphi_{sh}^{(i)} &= \frac{1}{l} \sum_{k=1}^l |H(\chi)_k - H(\hat{\chi}_{[i]})_k|^2, \\ \varphi_{dh}^{(i)} &= \frac{1}{l} \sum_{k=1}^l |H(\chi)_k - H(\kappa_{[i]})_k|^2, \end{aligned} \quad (2)$$

where $|\cdot|^2$ represents the Euclidean norm, l is the length of hash sequence, and $\varphi_{sh}^{(i)}$ and $\varphi_{dh}^{(i)}$ represent the hash distance of similar image pair and different image pair, respectively. The hash distance between original and similar image, and between similar and similar images should be small; on the other hand, the hash distance between original and different images should be large. So, as for the proposed scheme, $\varphi_{sh}^{(i)}$ should be as small as possible, and $\varphi_{dh}^{(i)}$ should be as larger as possible. In order to obtain useful loss value, the Sigmoid is used to normalize the value of MSE to $[0.5, 1]$, as shown in the following equation:

$$S(x) = \frac{1}{1 + e^{-x}}, \quad (3)$$

where $S(\cdot)$ is the function of Sigmoid, and the whole loss function is

$$\min \Gamma = \alpha_1 \frac{1}{n} \sum_{i=1}^n S(\varphi_{sh}^{(i)}) - \alpha_2 \frac{1}{n} \sum_{i=1}^n S(\varphi_{dh}^{(i)}), \quad (4)$$

$$\text{s.t. } \alpha_1, \alpha_2 \in [0, 1],$$

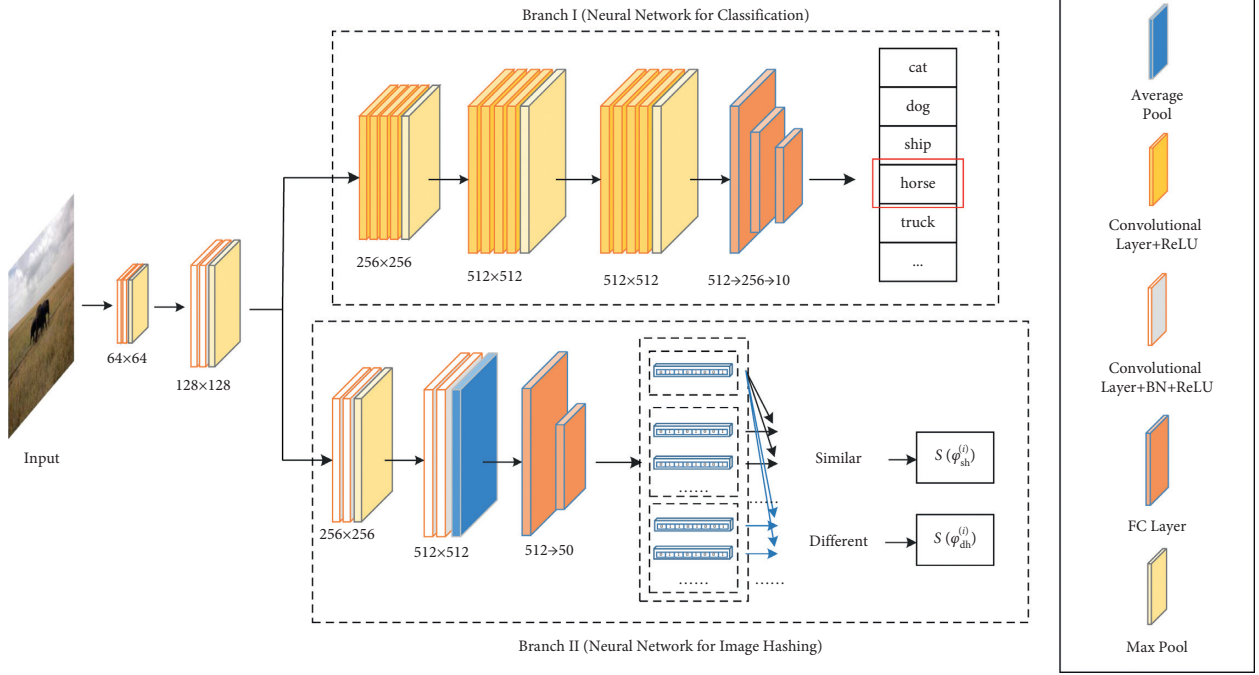


FIGURE 1: Structure of the dual-branch multitask neural network.

where Γ represents the whole loss function, and α_1 and α_2 both represent the super parameters, which are used to adjust the loss function. As for the pretrained model of VGG-19, the CEL (cross-entropy loss) function is used during the training. Denote a sample's tag *target* as $\{t_1, t_2, \dots, t_{10}\}$, the predicted output of the model is $\{o_1, o_2, \dots, o_{10}\}$, and then the value of CEL of the network is

$$L = - \sum_{i=1}^{10} t_i \log(o_i). \quad (5)$$

In particular, the following is used to measure and decide the best branch point among these max pooling layers:

$$\varsigma = \max_{j=1,2,\dots,m_v} \left\{ \frac{1}{n_v} \sum_{i=1}^{n_v} S(\varphi_{sh}^{(i)}) - \frac{1}{n_v} \sum_{i=1}^{n_v} S(\varphi_{dh}^{(i)}) \right\}, \quad (6)$$

where $\varphi_{sh}^{(i)}$ and $\varphi_{dh}^{(i)}$ represent the hash distance of perceptual similar pair and different image pair, respectively. The m_v is the quantity of batches in the test image set, and n_v is the quantity of similar images or different images in one batch. The smaller the ς , the more suitable the connection point is.

2.3. Training Strategy. Based on the pretrained model of VGG-19, we use the prepart of the network of it as the feature extractor, but the parameters of it are fixed, which will not be updated during the training of the hash sequence generation network. Note that, after the testing of equation (6), the second max pooling of VGG-19 is selected as the branch point, and the whole structure of the network and classification branch is shown in Table 1.

First, before the training of VGG-19, the CIFAR-10 training dataset is used in our scheme, and the optimizer is

SGD. In the training, the training momentum is 0.9, the weight attenuation is 5×10^{-4} , and according to the number of training epochs, the value of the learning rate is constantly adjusted to accelerate the convergence of the network and improve the accuracy of classification.

During the training and testing of the network, the super parameters α_1 and α_2 are both set as 1, at the same time, the optimizer *Adam* is used for the training of the proposed network (II branch), the learning rate is 0.001, and the value of epoch is 200.

The training dataset of the proposed neural network includes η original images, which are randomly selected from the COCO dataset [28]. Each original image would be transformed into 67 similar images through the attack operations given in Table 2, and we also added 67 different images with the original image. The original image, similar images, and different images have been combined into an image group, and therefore, we will obtain η image groups. Each time, we input an image group into proposed network for training.

3. Experimental Results and Comparisons

In order to evaluate the effectiveness and superiority of proposed scheme, the proposed neural network model in the aspects of perceptual robustness, discrimination, performance of content authentication, image classification, and computational complexity was tested and compared. The MSE is used to measure the hash distance as follows:

$$D(\mathbf{Q}^{(1)}, \mathbf{Q}^{(2)}) = \frac{1}{l} \sum_{j=1}^l |\mathbf{Q}_j^{(1)} - \mathbf{Q}_j^{(2)}|^2, \quad (7)$$

where l represents the length of the hash sequence, and $\mathbf{Q}_j^{(1)}$ and $\mathbf{Q}_j^{(2)}$ are j -th values of hash sequences $\mathbf{Q}^{(1)}$ and $\mathbf{Q}^{(2)}$,

TABLE 1: Structure of the multitask neural network.

Input (32×32 RGB image)		
	conv3-64 + ReLU	
	conv64-64 + ReLU	
	MaxPooling	
	Conv64-128 + ReLU	
	Conv128-128 + ReLU	
	MaxPooling	
Branch I		Branch II
Conv128-256 + ReLU		
Conv256-256 + ReLU		Conv128-256 + BN + ReLU
Conv256-256 + ReLU		Conv256-256 + BN + ReLU
Conv256-256 + ReLU		
MaxPooling		MaxPooling
Conv256-512 + ReLU		
Conv512-512 + ReLU		Conv256-512 + BN + ReLU
Conv512-512 + ReLU		Conv512-512 + BN + ReLU
Conv512-512 + ReLU		
MaxPooling		AvgPooling
Conv512-512 + ReLU		
Conv512-512 + ReLU		FC-512 + ReLU
Conv512-512 + ReLU		
Conv512-512 + ReLU		
MaxPooling		
FC-512 + ReLU		
FC-256 + ReLU		FC-50
FC-10		
Softmax		

TABLE 2: Eight common content retention operations (attack operations) for images.

Operations	Parameter name	Values of parameter
Speckle noise	Variance	0.01, 0.05, 0.1, 0.2, 0.3
Scaling	Scaling ratio	0.2, 0.3, 0.4, 0.5, 1.5, 2, 4
Median filtering	Filter size	1, 3, 5, 7, 9, 11, 13, 15, 17, 19
Circle average filtering	Filter size	1, 5, 10, 15, 20, 25, 30, 35, 40
JPEG compression	Quality factor	1, 5, 10, 30, 50, 70, 90, 95
Rotation and cropping	Rotation angle	1, 2, 3, 4, 5, 6, 8, 10, 12
Gamma correction	Γ	0.55, 0.65, 0.75, 0.85, 0.95, 1.05, 1.15, 1.25, 1.35, 1.45
Gaussian filtering	Variance	0.01, 0.02, 0.03, 0.04, 0.05, 0.1, 0.15, 0.2, 0.25

respectively. During the measurement of hash distance D , if D is smaller than threshold θ , the image pair are defined as similar images; on the contrary, they are defined as different images. Note that, the hardware environment of all experiments was uniform, and the CPU was i9-10900X, the GPU was RTX 2080 Ti, and the RAM was 32 GB.

3.1. Robustness Analysis. In the robustness test, 1,000 images (not in the training dataset) from the COCO dataset [28] were randomly sampled as original images for content retention attack operations. Common robust attacks included speckle noise, median filtering, and rotation and cropping.

Specific operations and parameter settings are shown in Table 2. Each image generated 67 perceptually similar images, and a total of 67,000 similar images were obtained. Meanwhile, 67,000 hash distances were obtained by using equation (7). Table 3 shows the extreme value, mean value, and standard deviation of each attack operation. The extreme max and min values are used to indicate the overall numerical range, and the mean and standard deviation values represent the situation of hash distance fluctuations. However, as the robustness testing data of all 1,000 images were difficult to display, five typical standard images, named *Airplane*, *Baboon*, *Boat*, *House*, and *Peppers*, were selected to display, as shown in Figure 2. After the content retention attack operations with eight different parameters as shown in Table 2 were used for these five images, the distance measurement was carried out according to the hash sequence generated from the original image and 67 similar images by using equation (7). Figure 3 shows the changes of 5×67 hash distances and demonstrates the superior robustness of the proposed scheme.

As shown in Figure 3, for JPEG compression, speckle noise, circle average filtering, median filtering, and scaling attack operations, the hash distances are small, less than 0.25. In the Gaussian filtering operation, with the increase of variance, the hash distance of *Boat* increases greatly compared with that of the other four images, but it is still in an acceptable range. Although the rotation and cropping and gamma correction operations have strong changes, compared with other attack operations, the average values of hash distances of these two operations are 0.3724 and 0.2089, respectively. As observed in Table 3, the performance of the proposed scheme was still excellent under rotation and cropping and gamma correction operations.

3.2. Discrimination Capability. To verify the discrimination capability of the proposed scheme, we used the UCID database [29], which contains 1,338 different images with sizes of 512×384 and 384×512 . First, we generated hash sequences of the first 1,000 images in [29]. Then, we calculated the hash distance D between each image and other 999 images. So that, we obtained $(1,000 \times 999)/2 = 495,500$ hash distances. Through the analysis of this experiment, and estimated according to values, the distribution of these hash distances followed the data distribution of mean $\mu = 1.871$ and standard deviation $\sigma = 2.258$. Obviously, the smaller the threshold θ , the lower the collision probability, which means better discrimination capability. When the hash distance D of two images is less than the preset threshold value θ , two images are defined as perceptual similar image pair. If the threshold value θ is too small, the network will distinguish some similar images as different ones, thus affecting the robustness performance of the proposed scheme. Therefore, we need to choose an appropriate threshold θ to achieve a balance between perceptual robustness and discrimination.

As can be seen from Table 3, the average hash distance of perceptually similar images in common image content retention operations is less than 0.4. In addition, when the threshold value $\theta = 0.4$, the collision probability of the

TABLE 3: Statistics of hash distances under image content retention operations.

Operations	Min.	Max.	Mean	Std.
Speckle noise	1.8×10^{-4}	0.1751	0.0253	0.0377
Scaling	8.0×10^{-5}	0.0550	0.0046	0.0109
Median filtering	0	0.0091	0.0026	0.0025
Circle average filtering	0	0.2417	0.0512	0.0646
JPEG compression	1.3×10^{-4}	0.1138	0.0131	0.0253
Rotation and cropping	1.6×10^{-3}	3.2829	0.3724	0.7473
Gamma correction	5.9×10^{-4}	2.7986	0.2089	0.5312
Gaussian filtering	7.9×10^{-4}	0.1424	0.0268	0.0321

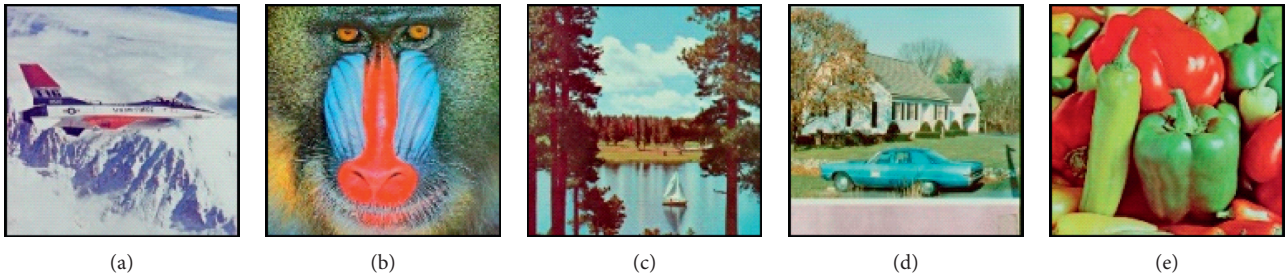


FIGURE 2: Five standard images for testing. (a) Airplane. (b) Baboon. (c) Boat. (d) House. (e) Peppers.

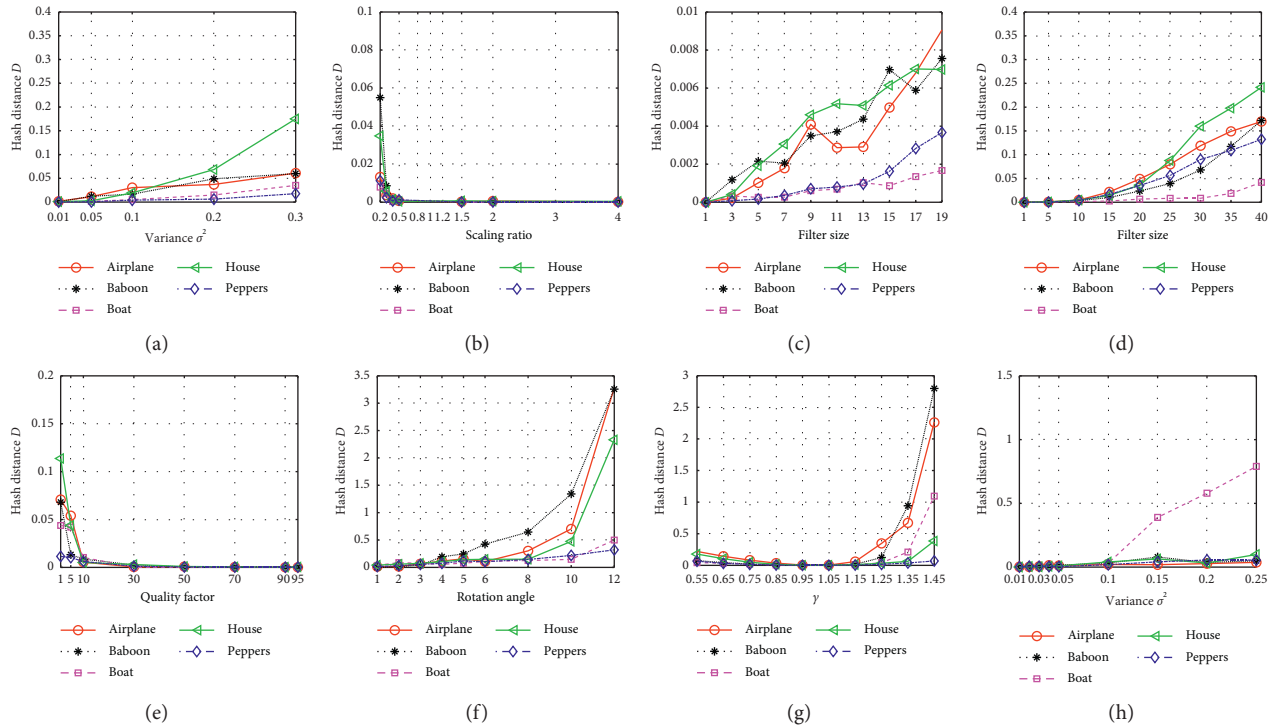


FIGURE 3: Test curves of five standard images in different image attack operations. (a) Speckle noise. (b) Scaling. (c) Median filtering. (d) Circle average filtering. (e) JPEG compression. (f) Rotation and cropping. (g) Gamma correction. (h) Gaussian filtering.

proposed scheme is 0.0607, which is shown in Table 4; that is, 93.93% of different images are correctly judged. Although hash distances between some perceptually similar images in Table 3 are greater than 0.4, they are only a very small part of

the results and have little impact on the overall performance. Therefore, in the proposed scheme, we set the threshold $\theta=0.4$ to achieve the balance of perceptual robustness and discrimination.

TABLE 4: The collision probability P_c under different threshold θ

Threshold θ	Collision probability P_c
0.90	0.2741
0.80	0.2284
0.70	0.1827
0.60	0.1385
0.50	0.0972
0.40	0.0607
0.30	0.0308

3.3. *Performance of Content Authentication.* In order to illustrate the superiority of the proposed scheme, the perceptual robustness and discrimination were considered, and we compared the proposed scheme with other four classical image hashing schemes: DCP [7], RP-IVD [4], RP-NMF [3], and DAE NN-based [24]. The first three schemes used traditional methods, and the last method used deep learning. Because the perceptual robustness and discrimination of the scheme are contradictory, when the perceptual robustness of the scheme is strong, the discrimination must be relatively weak, and vice versa. In order to compare the proposed scheme fairly with the other four schemes, we considered the combined performance of perceptual robustness and discrimination of each scheme as the content authentication ability to perceptually similar and different images.

In this experiment, we randomly selected 1,000 images (not in the training dataset) from [28] to construct a testing image dataset. Each image of this dataset corresponded to 67 perceptual similar images generated by the image content retention operations shown in Table 2 and 67 different images were randomly selected from this dataset. Through setting different values of θ , P_{FAR} and P_{FRR} can be calculated by the following equation:

$$\begin{aligned} P_{\text{FAR}}(\theta) &= \Pr(D(\mathbf{H}^{(1)}, \mathbf{H}^{(2)}) \leq \theta), \\ P_{\text{FRR}}(\theta) &= \Pr(D(\mathbf{H}^{(1)}, \mathbf{H}^{(2)}) > \theta), \end{aligned} \quad (8)$$

where P_{FAR} is the probability of the hash distance less than θ . On the contrary, P_{FRR} is the probability of the hash distance larger than θ , and $\Pr(\cdot)$ is the function of probability. F_1 score is an important quantitative index used to measure the accuracy of content authentication, and when the F_1 score is larger, the performance of content authentication is better. The calculated method of F_1 score is

$$F_1 = \max_{\theta} \left\{ \frac{2 \cdot [1 - P_{\text{FAR}}(\theta)] \cdot [1 - P_{\text{FRR}}(\theta)]}{[1 - P_{\text{FAR}}(\theta)] + [1 - P_{\text{FRR}}(\theta)]} \right\}. \quad (9)$$

Table 5 lists the comparison of F_1 scores of four schemes in different image content retention operations, and bold one of F_1 scores in each line is the best one. The proposed scheme is almost ahead of existing four excellent image hashing schemes in the eight attack operations, and the mean of F_1 score in our scheme is the largest among them, as shown in Table 5. In particular, compared with DAE NN-based [24], four F_1 scores of our scheme are significantly ahead, and other four F_1 scores are also very close. To further demonstrate the overall performance of content

authentication based on perceptual robustness and discrimination, we used ROC (receiver operating characteristic) to demonstrate the overall performance of our scheme and other four schemes. In Figure 4, the abscissa is P_{FRR} , the ordinate is $1 - P_{\text{FAR}}$, and the ROC curves are closer to top left corner, which means that the better performance of content authentication. Through ROC curves of four schemes in Figure 4, the curve of our scheme is closer to top left corner than other curves. Although the curve of the scheme [24] is close to ours, our scheme is better from the small graph magnified in the lower right. To sum up, according to the quantization results and ROC curves, our scheme had satisfactory performance than [3, 4, 7, 24] in image content authentication.

The generalization ability of our scheme was also tested, different image datasets were used, including COCO dataset [28], UCID dataset [29], Imagenet dataset [30], and NUS_WIDE dataset [31]. ROC curves generated by our scheme for different image datasets are shown in Figure 5. As observed from Figure 5, our scheme has good generalization ability in different image datasets, and at the same time, our scheme has satisfactory adaptive capacity.

3.4. *Performance of Image Classification.* Because the multitask neural network was used in the proposed scheme, apart from the function of hash authentication, the network also has the function of image classification. We used original images of CIFAR-10 dataset to test the accuracy rate of image classification in our pretrained VGG-19 neural network. There are 1,000 images in this testing dataset. Some results of the image classification of our pretrained network are displayed in Figure 6. The images of the first and second lines are completely correctly identified, but ship (as shown in the red box) is mistakenly identified as truck in the third line, which can be explained that ship had relatively similar characteristics to truck. The accuracy of classification in the final line is 90%. After the test of all images, the final accuracy rate was 93.42%, which means that our multitask neural network can do the task of image classification well.

3.5. *Computational Complexity.* In actual application, the performance of proposed scheme will face a variety of hardware resources and environment constraints. Therefore, considering the actual use and deployment of the scheme, it is necessary to test and compare the computational complexity of those schemes. In order to avoid the contingency, we randomly selected 100 images from [27] to generate the hash sequences and recorded the running time. Meanwhile, the same test would be carried out on other schemes to compare the advantages. The results are shown in Table 6, although the computational complexity and hash length are not best, considering the influence of multitask function and the tiny gap between ours and the best ones, our scheme is still the leading one in comprehensive performance.

3.6. *Application of Tampering Detection.* The tampering authentication is also important, and the relative test was also performed for the proposed scheme. Tampered images

TABLE 5: F_1 scores of different schemes in different content retention operations.

Operations	Proposed	DCP [7]	RP-IVD [4]	RP-NMF [3]	DAE NN-based [24]
Speckle noise	0.9996	0.9340	0.9409	0.9299	0.9995
Scaling	0.9996	0.8487	0.8909	0.9665	0.9993
Median filtering	0.9996	0.9631	0.9782	0.9891	0.9998
Circle average filtering	0.9988	0.9063	0.7121	0.7889	0.9895
JPEG compression	0.9993	0.7957	0.9383	0.9925	0.9998
Rotation and cropping	0.9990	0.6339	0.9936	0.9983	0.9961
Gamma correction	0.9987	0.9518	0.9903	0.8418	0.9999
Gaussian filtering	0.9991	0.9195	0.8148	0.8656	0.9992
Mean	0.9992	0.8708	0.9074	0.9216	0.9979

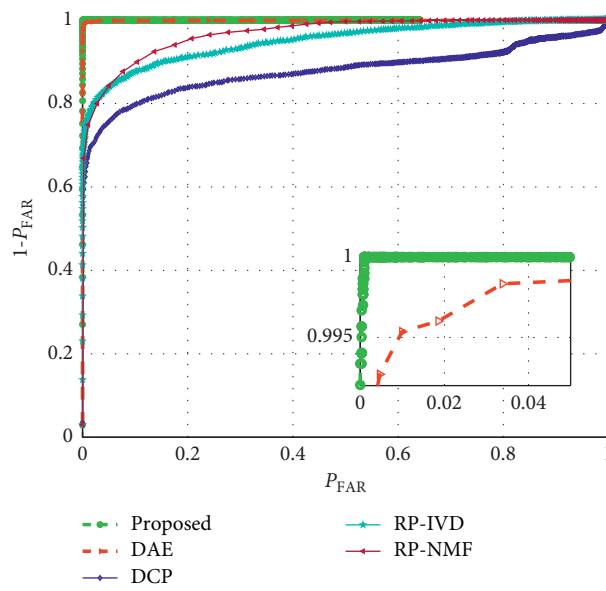


FIGURE 4: ROC curves about comprehensive performance of five schemes.

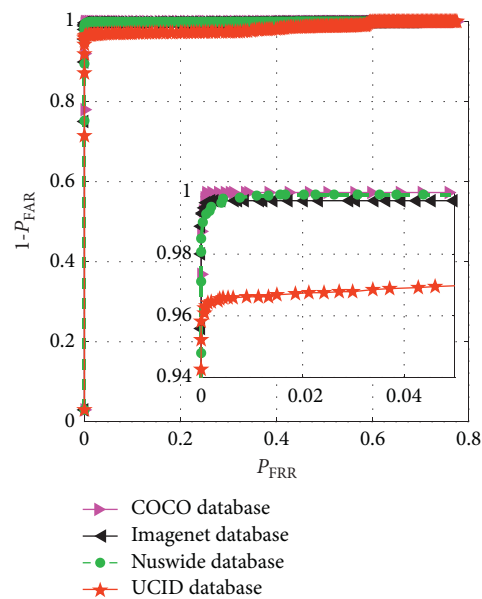


FIGURE 5: ROC curves of the proposed scheme in each image dataset.

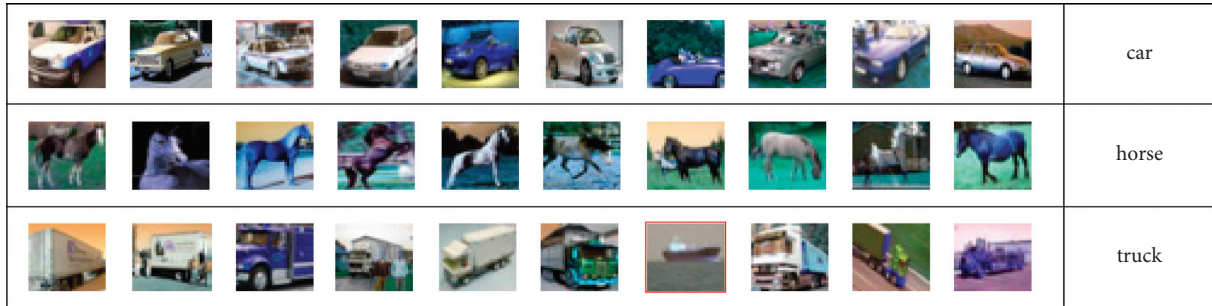


FIGURE 6: Partial results of image classification.

TABLE 6: The computational complexity and hash length of each scheme.

Indicator	Proposed	DCP [7]	RP-IVD [4]	RP-NMF [3]	DAE NN-based [24]
Computational complexity (ms)	48.5	154.62	435.2	645.2	17.9
Hash length (digits)	50	64	40	64	50
Multitask function	√	×	×	×	×

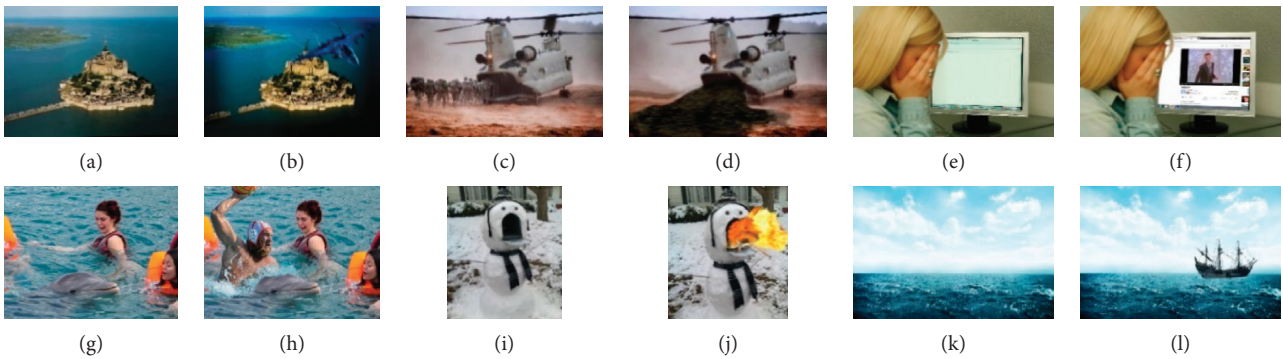


FIGURE 7: Six pairs of representative tampered test images in the IMD2020 database.

TABLE 7: Hash distance between the original image and corresponding tampered version.

Image pair	Hash distance D
(a) and (b)	1.36722
(c) and (d)	6.77353
(e) and (f)	1.5763
(g) and (h)	2.50353
(i) and (j)	1.02164
(k) and (l)	1.14891

can be judged by comparing the hash distance between original image and other images efficiently in a large image dataset. In order to test the tampering detection capability of proposed scheme, we randomly selected some original images and corresponding tampered versions from [32]. In Figure 7, we displayed six pair original-tampered images and, in each pair, original image in left, tampered image in right.

The hash distances of different tampered image pairs in Table 7 are larger than $\theta=0.4$, which indicates the hash sequences are significantly different between the original

image and corresponding tampered version. Therefore, the proposed scheme has good tampering authentication performance in practical application, and it can be applied to certain tampering detection scenarios.

4. Conclusion

In order to improve the performance of the image hashing scheme and the reusability of the neural network, we proposed a dual-branch multitask neural network with functions of hash sequence generation and image classification. By looking for the branch point in multiple max pooling layers of VGG-19 network and adding two branch networks, the proposed neural network could have two functions and used one feature extractor. In the two branch networks, one is the original network for image classification after the branch point of the VGG-19 network, and the other is the proposed network to generate the hash sequence. In order to ensure the network converges to the target result, a loss function is proposed to measure the hash distance, which is combined with the MSE and Sigmoid function. The experimental results

show that the proposed scheme has superior robustness and discrimination, and the testing results of image classification are better than the existing classical schemes. The proposed scheme can resist speckle noise, median filtering, rotation and cropping, etc., and it also has advantage in content authentication performance. Through the comparison of ROC curves with other schemes, it can be seen that the proposed scheme is still ahead of them in comprehensive performance. In addition, it has some superiority and applicability in computational complexity and tampering detection applications. In terms of task of image classification, it can be applied to common task of image classification to ensure the function of multitask and improve the applicability of the proposed network to multiple scenarios.

Data Availability

The image datasets used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grants 62172280, U20B2051, and 62172281, in part by the Natural Science Foundation of Shanghai under Grant 21ZR1444600, and in part by the STCSM Capability Construction Project for Shanghai Municipal Universities under Grant 20060502300.

References

- [1] M.. Schneider and S.-F. Chang, "A robust content based digital signature for image authentication," vol. 3, pp. 227–230, in *Proceedings of the 3rd IEEE International Conference on Image Processing*, vol. 3, pp. 227–230, IEEE, Lausanne, Switzerland, sep 1996.
- [2] C.-P. Yan, C.-M. Pun, and X.-C. Yuan, "Multi-scale image hashing using adaptive local feature extraction for robust tampering detection," *Signal Processing*, vol. 121, pp. 1–16, 2016.
- [3] Z. Tang, X. Zhang, and S. Zhang, "Robust perceptual image hashing based on ring partition and NMF," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 3, pp. 711–724, 2013.
- [4] Z. Tang, Z. Huang, X. Zhang, and H. Lao, "Robust image hashing with multidimensional scaling," *Signal Processing*, vol. 137, pp. 240–250, 2017.
- [5] Z. Tang, X. Zhang, X. Li, and S. Zhang, "Robust image hashing with ring partition and invariant vector distance," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 1, pp. 200–214, 2015.
- [6] X. Nie, X. Li, Y. Chai, C. Cui, X. Xi, and Y. Yin, "Robust image fingerprinting based on feature point relationship mining," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 6, pp. 1509–1523, 2018.
- [7] C. Qin, X. Chen, X. Luo, X. Zhang, and X. Sun, "Perceptual image hashing via dual-cross pattern encoding and salient structure detection," *Information Sciences*, vol. 423, pp. 284–302, 2018.
- [8] C. Qin, Y. Hu, H. Yao, X. Duan, and L. Gao, "Perceptual image hashing based on weber local binary pattern and color angle representation," *IEEE Access*, vol. 7, no. 45, pp. 460–471, 2019.
- [9] Q. Shen and Y. Zhao, "Perceptual hashing for color image based on color opponent component and quadtree structure," *Signal Processing*, vol. 166, 2020.
- [10] Z. Huang and S. Liu, "Perceptual hashing with visual content understanding for reduced-reference screen content image quality assessment[[]]," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 7, pp. 2808–2823, 2020.
- [11] X. Wang, M. A. Jianfeng, and Y. Miao, "Efficient privacy-preserving image retrieval scheme over outsourced data with multi-user," *Journal on Communications*, vol. 40, no. 2, pp. 31–39, 2019.
- [12] Y. Li, J. Ma, and M. Yinbin, "Encrypted image retrieval in multi-key settings based on edge computing," *Journal on Communications*, vol. 41, no. 4, pp. 14–26, 2020.
- [13] J. Fridrich and M. Goljan, "Robust hash functions for digital watermarking," in *Proceedings of the International Conference on Information Technology: Coding and Computing*, pp. 178–183, IEEE, Las Vegas, NV, USA, March 2000.
- [14] H. Zhang, H. Zhang, Q. Liu, and X. Niu, "Image perceptual hashing based on human visual system," *Acta Electronica Sinica*, vol. 36, no. 12A, pp. 31–34, 2008.
- [15] C. Qin, C.-C. Chang, and P.-L. Tsou, "Robust image hashing using non-uniform sampling in discrete Fourier domain," *Digital Signal Processing*, vol. 23, no. 2, pp. 578–585, 2013.
- [16] V. Patil and K. Tanuja, "Image hashing using DWT-CSLBP," *Journal of Computers*, vol. 14, no. 3, pp. 210–222, 2019.
- [17] S. Liu and Z. Huang, "Efficient image hashing with geometric invariant vector distance for copy detection," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 15, no. 4, pp. 1–22, 2019.
- [18] C. Qin, X. Chen, D. Ye, J. Wang, and X. Sun, "A novel image hashing scheme with perceptual robustness using block truncation coding," *Information Sciences*, vol. 361–362, pp. 84–99, 2016.
- [19] Y.-N. Li and P. Wang, "Robust image hashing based on low-rank and sparse decomposition," in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pp. 2154–2158, IEEE, Shanghai, China, March 2016.
- [20] Z. Han, M. Long, J. Wang, and Y. Cao, "Deep hashing network for efficient similarity retrieval," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, pp. 2415–2421, AAAI Press, Beijing, China, Feb 2016.
- [21] J. Zhang and Y. Peng, "Query-adaptive image retrieval by deep-weighted hashing," *IEEE Transactions on Multimedia*, vol. 20, no. 9, pp. 2400–2414, 2018.
- [22] F. Shen, Y. Xu, L. Liu, Y. Yang, Z. Huang, and H. T. Shen, "Unsupervised deep hashing with similarity-adaptive and discrete optimization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 12, pp. 3034–3044, 2018.
- [23] S. Cheng, H. Lai, L. Wang, and J. Qin, "A novel deep hashing method for fast image retrieval," *The Visual Computer*, vol. 35, no. 9, pp. 1255–1266, 2019.
- [24] Y. Li, D. Wang, and L. Tang, "Robust and secure image fingerprinting learned by neural network," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, pp. 362–375, 2019.

- [25] C. Qin, E. Liu, G. Feng, and X. Zhang, "Perceptual image hashing for content authentication based on convolutional neural network with multiple constraints," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 11, pp. 4523–4537, 2021.
- [26] B. Chen, W. Tan, G. Coatrieux, Y. Zheng, and Y.-Q. Shi, "A serial image copy-move forgery localization scheme with source/target distinguishment," *IEEE Transactions on Multimedia*, vol. 23, pp. 3506–3517, 2021.
- [27] B. Chen, X. Liu, Y. Zheng, G. Zhao, and Y.-Q. Shi, "A robust GAN-generated face detection method based on dual-color spaces and an improved Xception," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, pp. 3506–3517, 2021.
- [28] T.-Y. Lin, M. Maire, S. Belongie et al., "Microsoft coco: common objects in context," *Computer Vision - ECCV 2014*, vol. 8693, pp. 740–755, 2014.
- [29] G. Schaefer and M. Stich, "Ucid - an uncompressed color image database," *Proceedings of in Storage and Retrieval Methods and Applications for Multimedia*, pp. 472–480, SPIE, Nottingham, UK, 2004.
- [30] D. Jia, W. Dong, R. Socher, Li-J. Li, K. Li, and L. Fei-Fei, "Imagenet: a large-scale hierarchical image database," in *Proceedings of the Conference on Computer Vision & Pattern Recognition*, vol. 17, no. 5, pp. 248–255, IEEE, Miami, FL, USA, June 2009.
- [31] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y. Zheng, "NUS-WIDE: a real-world web image database from National University of Singapore," in *Proceedings of the International Conference on Image & Video Retrieval*, vol. 48, pp. 1–9, ACM, New York, NY, United States, July 2009.
- [32] N. Adam, B. Mahdian, and S. Saic, "A large-scale annotated dataset tailored for detecting manipulated images," in *Proceedings of the IEEE Winter Applications of Computer Vision Workshops*, pp. 71–80, IEEE, Snowmass, CO, USA, March 2020.