Hindawi
Security and Communication Networks
Volume 2023, Article ID 5349392, 16 pages
https://doi.org/10.1155/2023/5349392

Research Article

Vision Transformer-Based Video Hashing Retrieval for Tracing the Source of Fake Videos

Pengfei Pei,[1,2] **Xianfeng Zhao**,[1,2] **Jinchuan Li**,[1,2] **Yun Cao**,[1,2] **and Xuyuan Lai**[1,2]

[1]*State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100083, China*
[2]*School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100083, China*

Correspondence should be addressed to Xianfeng Zhao; zhaoxianfeng@iie.ac.cn

Received 27 July 2022; Revised 6 May 2023; Accepted 31 May 2023; Published 28 June 2023

Academic Editor: David Megias

With the increasing negative impact of fake videos on individuals and society, it is crucial to detect different types of forgeries. Existing forgery detection methods often output a probability value, which lacks interpretability and reliability. In this paper, we propose a source-tracing-based solution to find the original real video of a fake video, which can provide more reliable results in practical situations. However, directly applying retrieval methods to traceability tasks is infeasible since traceability tasks require finding the unique source video from a large number of real videos, while retrieval methods are typically used to find similar videos. In addition, training an effective hashing center to distinguish similar real videos is challenging. To address the above issues, we introduce a novel loss function, hash triplet loss, to capture fine-grained features with subtle differences. Extensive experiments show that our method outperforms state-of-the-art methods on multiple datasets of object removal (video inpainting), object addition (video splicing), and object swapping (face swapping), demonstrating excellent robustness and cross-dataset performance. The effectiveness of the hash triplet loss for nondifferentiable optimization problems is validated through experiments in similar video scenes.

1. Introduction

Video forgery has gained global attention, leading to increased focus on forgery detection [1–3]. Common techniques for video semantic editing include object removal (video inpainting), object addition (video splicing), and object swapping (face swapping) [4–7]. Malicious use of these technologies can cause harm to individuals and organizations, and fake videos can have serious consequences for politics, society, finance, and the law. Current methods output probability values but lack interpretability and have limitations in real-world applications [6, 8, 9]. Moreover, existing forgery detection methods perform poorly on independent testing and have poor robustness to common video processing techniques used on the Internet [6]. Therefore, a reliable and robust forgery detection method is essential.

Inspired by hash retrieval, we propose a hash-based source-tracing method. However, the discrete distribution of the hash space and the nonsmooth calculation function using the Hamming distance result in nondifferentiable optimization problems. Traditional hash retrieval is usually employed to find similar videos within the same category, where videos of different categories have significant semantic differences, making it easy to train different hash centers. However, the challenge in source tracing lies in the fact that videos in the dataset may be similar and that their initial hash codes are difficult to differentiate, which makes it hard to train hash centers with significant differences. To address these challenges, we introduce a new loss function called the hash triplet loss, which replaces the Hamming distance calculation function with a differentiable function implemented in PyTorch. The hash triplet loss can iteratively optimize hash codes, gradually differentiating videos with

subtle differences, even when the differences are not immediately apparent.

Figure 1 illustrates the approach for learning hash codes for triplet-based retrieval and tracing. The hash retrieval methods are based on triplets (q, q^+, q^-), where (q, q^+) is a positive sample and q^- is a negative sample [10–12]. The method increases the distance between (q, q^+) and q^- within a triplet, decreases the distance between q and q^+, and learns the local similarity between elements within the triplet.

Instead, we treat $\mathbb{X} \in \{s_x, f_{x1}, f_{x2}, \ldots, f_{xn}\}$ as one class and train a hash center Center$X$ for a real video s_x and its associated fake videos f_{xi}. The hash triplet loss is based on triplets (s_x, f_{xi}, f_{xj}), where s_x is the real video and f_{xi} and f_{xj} are two randomly selected relevant fake videos. In each training iteration, the hash triplet loss increases the distance between the hash codes of different-class triplets and decreases the distance between the hash codes of the fake videos (f_{xi}, f_{xj}) and the real video s_x in the same triplet. It learns the global similarity of a class of data.

Since each triplet always includes the real video s_x, the fake videos eventually generate a hash center around the real video. Therefore, our method reduces the reliance on a limited set of forged videos. By not relying on these forged traces, our method can improve the robustness of detecting forged videos against various processing techniques and the generalization of various forgeries. Ultimately, the hash centers of different classes are far apart, and the hash codes of videos from the same class are clustered around their corresponding hash center.

As shown in Figure 2, the distribution of video hash codes is presented at different stages. Initially, the binary hash codes of real and fake videos in the dataset are mixed together, making them difficult to distinguish. During training, the hash codes of the real and related fake videos gradually converge, while the hash codes of unrelated videos separate. Eventually, a hash center is trained for each real video and its related fake ones, and the average Hamming distance of each hash center is close to half of the length of the hash code. The generated hash centers are close to the optimal hash distribution [13].

We use the pyramid vision transformer (PVT) v2 [14] as the backbone for feature extraction. PVT v2 is an effective network for learning image recognition features based on the vision transformer (ViT) architecture. To better capture the temporal information of videos, we design a temporal encoding module that is commonly used in the ViT structure [15]. We first train the network model and hash centers using the hash triplet loss. Then, we calculate the hash center with the minimum Hamming distance for the given hash code of the detected video and find the related real video through the index of the hash center. Finally, we use human-level comparison to judge the difference between the real and detected videos to determine whether the detected video is fake. When the found real video is not related to the detected video, detection fails. In summary, the contributions of this paper are as follows:

(i) Our method offers a more reliable alternative to probability-based detection techniques, making it a promising solution for real-world applications, particularly in critical scenarios involving individuals.

(ii) We have designed a novel loss function, hash triplet loss, for forgery detection through source tracing. Extensive experimental results have demonstrated that our method outperforms the state-of-the-art forgery detection methods. Our code and models have been released on GitHub and have received considerable attention.

(iii) Our method does not rely on potential forgery artifacts, thereby improving the robustness and generalization of detection. We conducted extensive experiments on multiple datasets of three different types, demonstrating the effectiveness of our approach for detecting various types of synthetic forgeries, such as DeepFake, video splicing, and video inpainting.

2. Related Works

2.1. DeepFake Detection. DeepFake detection methods typically output probability values [16–19]. Some learning-based methods directly learn fake features from data without relying on any manual features [20–22], while others attempt to improve the interpretability of detection by labeling fake traces [23–26]. Audio-based DeepFake detection methods [24, 25] detect fake videos by using audio information. FakeLocator [26] detects full-resolution facial fake videos by generating corresponding grayscale images using GAN-generated facial intrinsic defects. Find-X [23] uses unsupervised learning to learn potential inconsistent fake features and outputs visualized fake trace results, thereby improving the generalization ability of fake detection. ISTVT [3] proposes an interpretable spatiotemporal video transformer for capturing spatial fake traces and temporal inconsistencies, achieving strong DeepFake detection.

2.2. Video Inpainting Detection. Object inpainting has been widely applied in real-world applications such as object removal [27–29]. Methods based on 3D CNNs have shown poor performance in video inpainting. Recently, flow-based approaches have incorporated optical flow into networks used for video inpainting [30, 31]. This alleviates the time issue of video inpainting but inevitably leaves temporal artifacts in the generated results. Several works have been proposed for video inpainting localization recently. Learning-based inpainting localization methods aim to extract semantic representations through a large amount of training data [32, 33]. However, the performance of these methods sharply declines on new datasets due to their reliance on large training datasets. Others apply advanced features to enhance robustness. VIDNet [9] uses LSTM-based ELA and temporal structures to localize video inpainting. HPF [34] explores high-pass filtering to distinguish high-frequency noise and fake images. FAST [8]

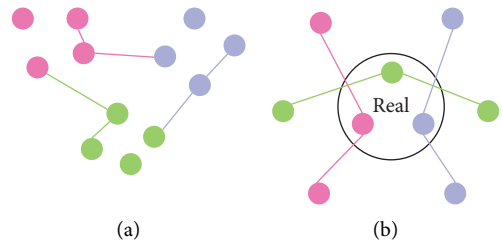

Figure 1: Illustration of traditional retrieval-based triplet and trace-based triplet learning for hash codes: (a) retrieval-based triplet; (b) source-tracing triplet.

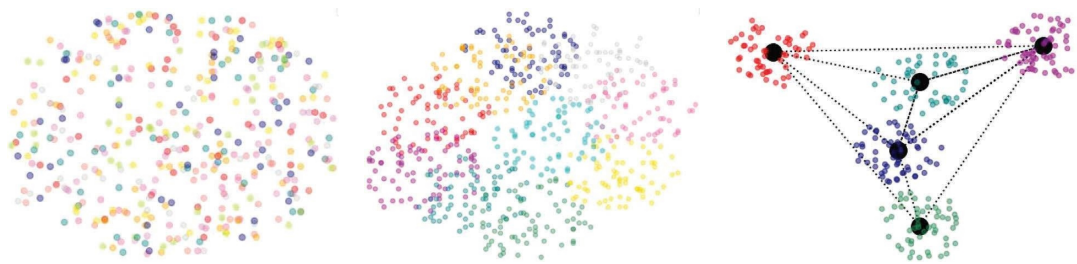

Figure 2: The distribution of learned hash centers during training. (a) Prior to training, the hash distribution of the dataset is scattered. (b) As training progresses, the hash distribution of each group in the dataset gradually clusters, resulting in changing hash centers for the dataset. (c) Eventually, the hash distribution of the dataset becomes sparse, the Hamming distance between the hash distributions of each group of data becomes very small, and the average Hamming distance between groups approaches half of the hash bits. The hash centers (represented by black points) are far apart from each other, and each group of data is located around a hash center: (a) raw, (b) training, and (c) final.

combines frequency-domain characteristics and temporal ViT to improve the performance of video inpainting localization. However, these methods do not consider the inherent artifacts of the inpainting manipulation process, making them ineffective when a new forgery method is proposed.

2.3. Video Splicing Detection. Since splicing is a relatively simple task, image/video splicing is usually performed manually with tools such as Photoshop. Due to the lack of video splicing datasets, there have been few studies on video splicing detection. Image splicing can be detected at the pixel level. PQMECNet [35] uses the local estimation of the JPEG quantization matrix to distinguish spliced regions taken from different sources. MVSS-Net [6] learns semantic-agnostic and more generalizable features by utilizing noise distribution and boundary artifacts around tampered regions. ComNet [36] is customized to approximate JPEG compression operation, thereby improving performance against adversarial JPEG compression. The challenge of splicing localization is to improve the robustness against various postprocessing operations [6] such as compression and blur.

2.4. Hash Retrieval. Hash retrieval methods map high-dimensional content features of images or videos to Hamming space (binary space), reducing the memory space requirements in image or video retrieval systems, improving retrieval speed, and meeting the requirements for massive data retrieval [10, 12, 37, 38]. Retrieval methods based on image similarity matching are computationally expensive and time-consuming, as they require matching a large number of key frames in videos [12, 37]. Changes in the semantics of fake videos are more obvious and significantly affect the matching accuracy. In contrast, hash-based retrieval methods are faster and require fewer resources, and their accuracy mainly depends on the quality of the hash centers [11, 13]. Traditional triplet learning methods use (q, q^+, q^-), capturing only local data similarity from two or three samples and ignoring global data similarity [10, 12]. Subregion [11] proposed a novel subregion localized hashing approach to learn compact within-class and large between-class hash codes that capture fine-grained local information for efficient fine-grained image retrieval. DLTH [12] introduced a new method for generating triplets from a knowledge distillation module to introduce more triplets during training and proposed a new listwise triplet loss to capture relative similarities in the new triplets. Due to the differences in processing logic, directly applying existing hash retrieval algorithms to source tracing is inappropriate.

2.5. Source-Tracing Detection. In recent years, the method of detecting fake data through source tracing has gradually attracted researchers' attention. These methods typically retrieve the source of the data under test from an existing real database and then judge the authenticity by manually

comparing the differences between the data under test and the real data. Shang et al. [39] use distributed blockchain technology to trace the source of fake news, which can effectively prevent the spread of fake news and provide reliable fake news detection. Dwivedi et al. [40] propose a social media framework based on blockchain and watermarking to control the spread of fake news. It helps to reduce the spread of fake news by tracing the root or source of fake news on social media. Shrivastava et al. [41] propose a model to investigate the spread of fake news related to the COVID-19 pandemic, thereby alleviating the pressure on online social network users. Zhu et al. [42] propose a voice antifraud method. The experimental results on the ASVspoof 2019 LA dataset show that the proposed method achieves a 20% performance improvement compared to traditional binary deception detection methods. The methods related to news and voices demonstrate that using source-tracing detection methods is not only effective but also highly applicable to real-world scenarios in the industry.

2.6. Vision Transformer. Currently, networks based on ViT have achieved great success in various fields, including image and video tasks [15, 43–46]. ViT is an effective structure for feature extraction from sequential data, making it particularly suitable for extracting temporal features from videos [14, 47, 48]. In addition to the classic 3D CNN and hybrid 2D CNN architectures, ViT provides an alternative solution for video understanding tasks. ViViT [15] first proposed a pure ViT-based structure for video classification, which uses token temporal and positional encodings to more effectively extract spatiotemporal features from videos. In early research, pure ViT-based structures required larger datasets and more memory consumption compared to CNN models. HRFormer [43] improves memory and computational efficiency by utilizing a multiresolution parallel design introduced in high-resolution convolutional networks, as well as local window self-attention conducted on small non-overlapping image windows. Recent studies have combined CNN and ViT to achieve better performance [49]. PoolFormer [44] improves the self-attention mechanism-based ViT structure into a hybrid structure of CNN and ViT, significantly reducing computation consumption. With the evolution of deep-learning architectures, the hybrid architecture of CNN and ViT is a popular choice. PVT v1 [48] inherits the advantages of both CNN and ViT and replaces the CNN backbone to make it a unified backbone in various visual tasks. It uses a progressive shrinking pyramid to reduce the computation consumption of large feature maps, achieving better performance in multiple tasks [48]. PVT v2 [14] reduces the computational complexity of PVT v1 to linear and significantly improves basic visual tasks such as classification, detection, and segmentation. In this paper, we use PVT v2 as the detection backbone and leverage token temporal encoding combined with PVT v2 for more effective video feature learning, given that PVT v2 is an image task model.

3. Method

In this section, we describe the complete procedure of our approach. As shown in Figure 3, our method involves three main stages: data preprocessing, hash center learning, and fake video source tracing. Initially, we restructure the dataset videos to adapt them to the training of the hash triplet loss. Next, we employ the hash triplet loss to learn the hash centers gradually and dynamically. Finally, we save the trained model and hash center and use the hash code of the fake video to trace the corresponding real video.

3.1. Data Preprocessing

$$\mathcal{O} = \begin{Bmatrix} s_{a1} & f_{a1} & f_{a2} & \cdots & f_{an} \\ s_{b1} & f_{b1} & f_{b2} & \cdots & f_{bn} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ s_{x1} & f_{x1} & f_{x2} & \cdots & f_{xn} \end{Bmatrix}. \tag{1}$$

The data preprocessing step reorganizes and combines the dataset in a way that is suitable for training our method with the hash triplet loss. Each subclip in the video is used to train the hash center so that the original real video can be accurately traced to detect any tampered videos based on any subclip. Given a dataset defined as $\mathcal{O}$ in equation (1), we partition each $\mathbb{X} \in \{s_x, f_{x1}, f_{x2}, \ldots, f_{xn}\}$ into a class and train a hash center c_x for each class of data $\mathbb{X}$. During training, we form triplets $\mathbb{U} = \{s_x, f_{xi}, f_{xj}\}$, where s_x is the real video and (f_{xi}, f_{xj}) are two randomly selected fake videos. To train independent hash centers for forgeries, each triplet unit always contains the original real video s_x. We recommend a triplet unit size of $\mathbb{U} > 8$ to ensure a uniform and reasonable distribution of data across different classes. It should be noted that data preprocessing is only applied during the training phase.

3.2. Hash Center Learning

3.2.1. Definition of Hash Triplet Loss. Inspired by the K-means clustering algorithm, the process of training the hash center is similar to clustering. It involves gradually adjusting the hash center to cluster real videos and related fake videos of the same class. The main idea of implementing the hash triplet loss is to increase the interclass loss and reduce the intraclass loss. The interclass loss refers to the Hamming distance between hash codes of videos from different classes, while the intraclass loss refers to the Hamming distance between hash codes of a real video and its related fake videos. The process of computing the complete hash triplet loss is illustrated in Algorithm 1. We input the hash codes and labels HL_s of the training instances, along with the associated hash center HC_s. Subsequently, we compute the intraclass loss and interclass loss using the function defined in Algorithm 1. The mathematical expression of $\mathscr{L}$ is defined as follows:

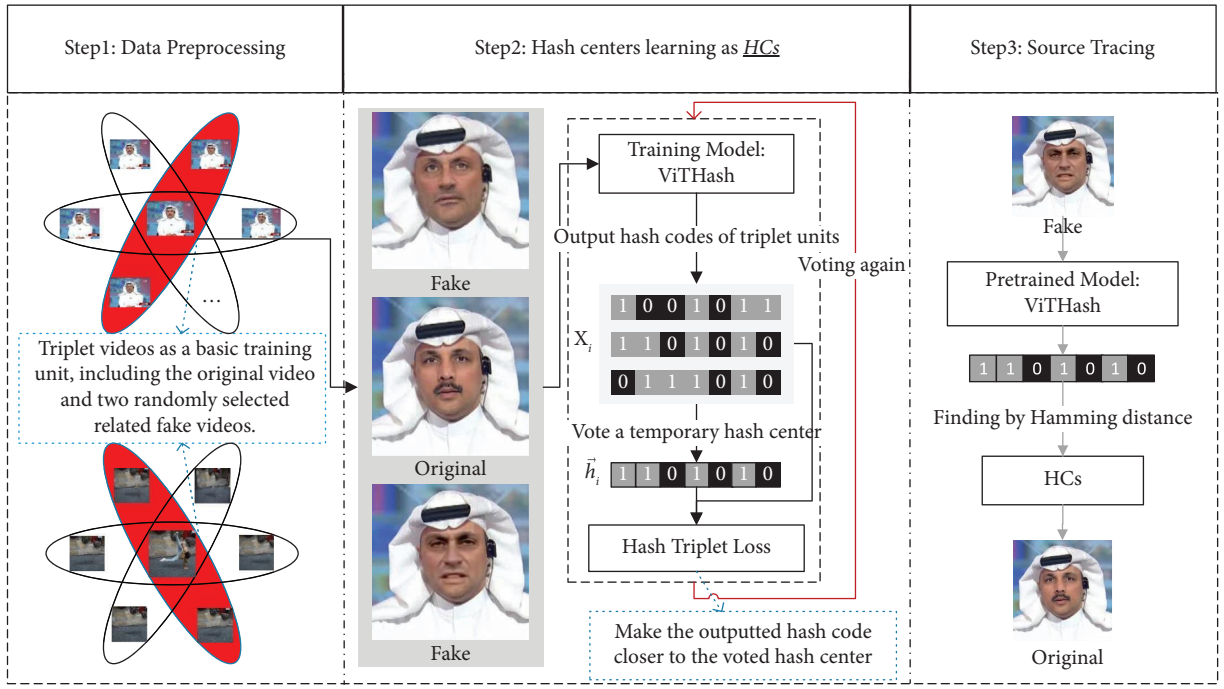

FIGURE 3: The complete process of training hash centers and performing source tracing can be divided into three steps: Step 1: data processing: the data are organized into a format suitable for training with the hash triplet loss. Step 2: hash center learning: the hash centers are dynamically trained, and a temporary hash center is generated by voting during each training iteration. Step 3: source tracing: the trained model and hash centers are utilized to search for the real video with the smallest Hamming distance from the detected video.

(1) ViTHash outputted hashes and related labels as HL_s;
(2) Voted hash centers and related labels as HC_s loss of **hash triplet loss**
(3) Calculate the intraloss between the triplet samples and the voted hash center;
(4) **Def** *IntraLoss*($\vec{h}$,$\overrightarrow{HC_i}$):
(5) **return** *mean*($|\vec{h} - \overrightarrow{HC_i}|$);
(6) ;
(7) Calculate the interloss between the triplet samples with the other hash centers;
(8) **Def** *InterLoss*($\vec{h}$,$\overrightarrow{HC_i}$)**:**
(9) **return** 1 − mean($|\vec{h} - \overrightarrow{HC_i}|$);
(10) ;
(11) Calculate the hash triplet loss;
(12) **Function** *Main* **:**
(13) $inter, intra, n, m = 0, 0, 0, 0$;
(14) **for** $label_h$, $\vec{h}$ ***in*** HL_s **do**
(15) **for** $label_o$, $\vec{o}$ ***in*** HC_s **do**
(16) **if** $label_t == label_v$ **then**
(17) $m+ = 1$;
(18) $intra+ = IntraLoss(\vec{h}, \vec{o})$;
(19) **else**
(20) $n+ = 1$;
(21) $inter+ = InterLoss(\vec{h}, \vec{o})$;
(22) ;
(23) ;
(24) **return** $intra \div m + intra \div n$;

ALGORITHM 1: The whole calculation process of the hash triplet loss.

$$\mathrm{dis}(\overrightarrow{x}, \overrightarrow{y}) = |\overrightarrow{x} - \overrightarrow{y}|, \tag{2}$$

$$\mathscr{L} = \sum_{i=1}^{m} \frac{\mathrm{dis}\left(\overrightarrow{v_i}, \overrightarrow{h_i}\right)}{m} + \sum_{j=1}^{n} \frac{\left(1 - \mathrm{dis}\left(\overrightarrow{v_j}, \overrightarrow{h_j}\right)\right)}{n}, \tag{3}$$

where m is the number of videos in the same class (intraclass) and n is the number of videos in different classes (interclass).

3.2.2. Voting Temporary Hash Centers. During each training iteration, given a triplet unit $\mathbb{U} = \{s_n, f_{nj}, f_{nk}\}$, the ViTHash network outputs the corresponding hash codes. Each triplet $\mathbb{U}$ votes to generate a temporary hash center $\overrightarrow{h}$ using $\mathrm{vote}(\mathbb{U}) = [\sum_{i=1}^{n} \mathbb{U}_i/n > 0.5]$, where $\mathbb{U}_i$ represents the ith column element of the matrix $\mathbb{U}$. The output is 1 if the mean of $\mathbb{U}_i$ is greater than 0, and 0 otherwise. The voting method for the temporary hash center is similar to the following:

$$\begin{bmatrix} 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 \end{bmatrix} \Longrightarrow \begin{bmatrix} 1 & 1 & 0 & 1 & 0 \end{bmatrix}. \tag{4}$$

The hash codes of the same triplet are encouraged to be close to this temporary hash center through the intraclass loss, while the hash centers of different triplets are pushed far away from each other through the *interclass* loss, implemented using equation (3). Through repeated iterative training and optimization, the temporary hash center gradually approaches the optimal hash center with an average Hamming distance close to half of the hash code length [13]. The trained model and hash center file are saved for future use.

3.2.3. Nondifferentiable Optimization for Similar Videos. Learning optimal hash centers through the network is challenging due to the high similarity of hash codes among similar videos. Nondifferentiable optimization is often used in deep learning-based hash code generation due to nonsmooth similarity metrics like Hamming distance, which can be solved using subdifferentials. For a function $f: \mathbb{R}^n \longrightarrow \mathbb{R}$, its subdifferential is defined as follows:

$$\partial f(x) = v \in \mathbb{R}^n: \ f(y) \geq f(x) + \langle v, y - x \rangle, \forall y \in \mathbb{R}^n, \tag{5}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product and $\partial f(x)$ represents the subdifferential of f at point x. The nonsmooth optimization problem can be written as $\min_{x \in \mathbb{R}^n} f(x)$, where $f(x)$ is a nonsmooth function.

Subgradient methods can be used to solve nonsmooth optimization problems, such as learning optimal hash centers. Specifically, the subgradient of the similarity metric with respect to the hash codes can be computed and used to update the hash codes. The update rule of subgradient methods is as follows:

$$h_{t+1} = h_t - \eta_t \cdot \partial S(h_t), \tag{6}$$

where h_t denotes the hash codes at iteration t, $S(h_t)$ denotes the similarity metric, η_t denotes the learning rate, and $\partial S(h_t)$ denotes the subgradient of the similarity metric at h_t.

In our PyTorch implementation of the hash triplet loss, we calculate the vector distance using mean(|torch.sub($\overrightarrow{h}$, $\overrightarrow{c}$)|) instead of the nonsmooth Hamming distance. The Hamming distance measures the similarity between two hash codes of the same length by counting the number of differing bits. The smaller the difference, the higher the similarity. Our implementation measures similarity using the average absolute difference between two vectors, where smaller values indicate higher similarity to the hash center. Therefore, in theory, these two methods can be used interchangeably.

In practice, we have observed that even for very similar videos, there exist slight differences in their hash codes. Increasing the length of hash codes (to 512 bits) allows for optimization of different bit elements and better representation of the slight differences. The interclass loss increases the Hamming distance between hash codes of different classes during each iteration to train optimal hash centers.

3.3. Fake Video Source Tracing. After training the model and hash centers, source tracing becomes a straightforward task, but it requires human-level interaction to judge whether the detected video is forged. Once $\mathbb{HC}$ is trained, we load both $\mathbb{HC}$ and the trained model. Given a detected video f and the hash code $\overrightarrow{h}$ outputted by the trained model, we calculate the Hamming distance between $\overrightarrow{h}$ and all hash centers HC*s*. We find the HC_i with the minimum Hamming distance to $\overrightarrow{h}$, along with its corresponding label. We use the label to retrieve the original genuine video s_i. Finally, we compare the detected video f with the genuine video s_i by human-level judgement. Since the tampered videos always have obvious semantic modifications, it is easy to distinguish the difference between the detected video f and the genuine video s_i. Thus, one can judge whether it is forged through human-level interaction.

4. Networks

In this section, we introduce the network architecture of our approach, as well as some advantages of our method.

4.1. Overview of Networks

4.1.1. ViTHash. As shown in Figure 4, ViTHash is used to train the hash center and trace the source. The feature extraction of ViTHash consists of a series of spatiotemporal PVT v2 [14] and multiple attention blocks. The first module, spatial transformer, focuses on spatial features, while the second module, temporal transformer, focuses on temporal features. Finally, the output is generated through the tanh function and then converted into binary codes using the sign() function in equation (8), where k represents the number of hash bits.

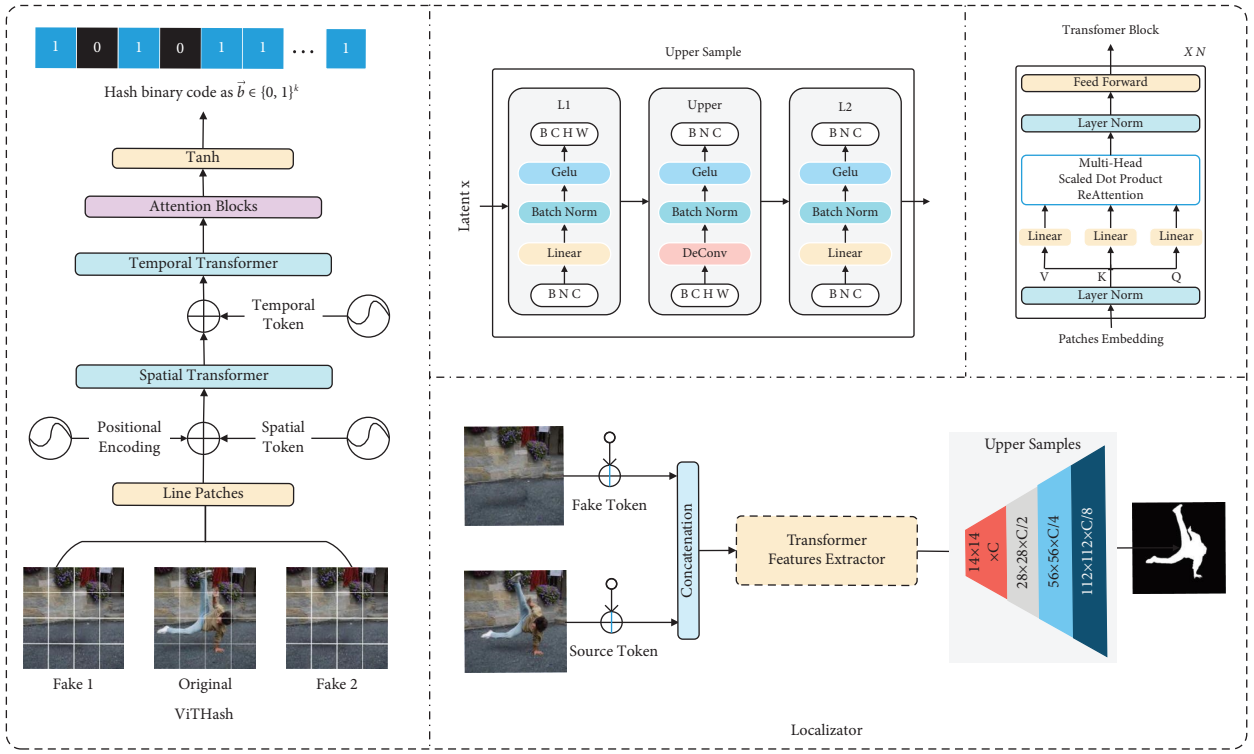

FIGURE 4: Overview of our proposed networks. Our method consists of two networks: ViTHash and localizator, and two basic modules: upper sampling and transformer block. ViTHash and localizator are composed of these basic modules. ViTHash trains hash centers from triplet videos, which include the original video and two randomly related fake videos. The trained hash centers are used to trace the source of fake videos. The localizator is designed to analyze the differences between the traced video and the fake video, which are not affected by the video quality or cropping. The different areas of the two videos are represented by generated masks.

$$\text{sign}(x) = \begin{cases} 1, & x \geq 0, \\ -1, & x < 0, \end{cases} \tag{7}$$

$$\overrightarrow{b} = \frac{(\text{sign}(x) + 1)}{2} \in \{0, 1\}^k. \tag{8}$$

4.1.2. Localizator. As shown in Figure 4, the localizator architecture is designed to facilitate comparison between real and fake videos. It serves as an auxiliary comparison network that outputs suspicious areas in grayscale, helping us distinguish the differences between the tracked and detected videos. We observed that the ViT-based network disrupted the spatial continuity of pixels when trained on linear patch images [45]. CNN blocks excel in learning high-level features and focusing on the correlation of local pixels, while ViT focuses on the long-range context and temporal dimension features of videos. To improve performance, we designed a hybrid CNN-ViT structure. In addition, we used an upper sampling module to gain more detailed insights into the differences in the regions of interest.

4.2. Advantages of the Proposed Method

4.2.1. Fast and Space Efficiency. We assume that the time required for detecting using different backbone network methods is relatively similar and denoted as t. For the traditional forgery detection method, the time cost is $t_1 = t$. The hashing retrieval method requires a time cost of $t_2 = \lambda + t \approx t$, where λ is the time needed to calculate the Hamming distance. In contrast, the content matching retrieval method takes a time cost of $t_3 = n \times t$, where n denotes the number of matching videos. In addition, hashing retrieval requires minimal storage space to store the hash code and video index. The hash code is a fixed-length binary bit (k bits), and the index is represented by a 32 -bit integer. The total storage space required is $(32 + k) \times n$, where n is the number of original videos. The scalability of hashing methods enables them to handle large datasets with ease, making them ideal for use in big data applications.

4.2.2. Better Versatility. ViTHash is a detection method that does not rely on forgery techniques and is forgery-independent, which makes it more versatile. Extensive experiments on multiple types of video forgery datasets have shown that ViTHash outperforms other methods in various types of forgeries. To ensure the validity and fairness of our results, we have made our experimental data and code publicly available.

4.2.3. Reliability. Traditional forgery detection methods detect videos by outputting probability values for detection. However, these values lack interpretability and cannot

provide fully reliable results, even when claiming to provide additional interpretable visual features. In critical scenarios involving high-profile individuals in government, military, and business, it is difficult to eliminate the impact of public opinion without conclusive and reliable evidence. Establishing a database of related real videos for these individuals can help trace malicious tampering based on these videos back to the original real videos. Comparing these original real videos with the fake videos provides reliable evidence for tampering detection.

5. Experiments

5.1. Experiment Setup. We conduct two sets of experiments: the ViTHash detection performance evaluation and the localizator evaluation. Evaluation of ViTHash six evaluation experiments and one ablation study is carried out. The six evaluation experiments include a DeepFake comparison experiment, a video tampering experiment, a video splicing experiment, a robustness experiment, a cross-dataset generalization experiment, and a similar-scene performance experiment. The evaluation of ViTHash detection performance using detection accuracy (ACC) as the evaluation metric is carried out. The localizator serves as an auxiliary comparison tool to facilitate the comparison of two videos. The evaluation of the localizator experiment outputs the pixel-level suspect region between two known methods. The localizator evaluation using mean intersection over union (mIoU) as the evaluation metric is carried out.

5.1.1. Implementation. Our model is implemented using PyTorch, and the code is released on GitHub. We use ffmpeg to extract frames from videos and train the model using a single NVIDIA RTX 3090 24GB GPU. Each model is trained for 2–5 epochs on the dataset. In addition, we use the adaptive moment estimation (ADAM) optimizer with a learning rate of $1e-5$. ADAM is computationally efficient, requires less memory, and performs well on large datasets.

5.1.2. Baseline Methods. In the ViTHash comparative experiment, we use accuracy as the evaluation metric. As the necessary implementation codes were not available, we cite the experimental results of the compared methods from the relevant papers. Compared to the existing forgery detection methods that directly output binary classification results, our method utilizes traceability to determine the authenticity. For fairness, we use the Top-1 retrieval accuracy as the evaluation metric because there is only one correct result for traceability. To evaluate the cross-dataset generalization performance, we compare Xception [50], HRNet [51], Face X-ray [52], ADD [1], and Grad-CAM [53] on the five subdatasets of FaceForensics++.

For the DeepFake comparison experiment, we select six methods: Xception [54], Face X-ray [52], Grad-CAM [53], STIL [55], ISTVT [3], and MRL [56] and compare them on Celeb-DF, DeepFakeDetection, and FaceForensics++ datasets. We also conduct a fine-grained comparison experiment with eleven methods: Xception [54], I3D [57], LSTM [58], TEI [59], ADDNet-3d [60], S-MIL [61], S-MIL-T [61], STIL [55], VTN [61], and ISTVT [3] on the FaceForensics++ dataset.

For the comparison experiment in the localizator, we chose mIoU as the evaluation metric and compared it with two known methods, DMAC [62] and DMVN [63].

5.2. Datasets

5.2.1. DeepFake Dataset. We evaluate our method on several publicly available datasets in the field of DeepFake detection. The FaceForensics++ (FF++) dataset [50] includes 1,000 real videos and 5,000 unique fake videos collected from YouTube. The Google/Jigsaw DeepFakeDetection (DFD) dataset [64] contains 363 original videos from 28 consenting actors and 3,068 fake videos. The Celeb-DF [65] dataset, which is part of the deep fake detection challenge, consists of 590 original videos and 5,639 fake videos.

5.2.2. Similar Scene Video Dataset. We create a dataset called DeepFake of similar scenes (DFS) to evaluate the detection performance of the hash triplet loss on similar videos, as shown in Figure 5(a). DFS aims to simulate scenarios like news conferences, which are highly similar and thus challenging to detect. We paid 75 actors to shoot similar-scene videos where they sit in front of the camera and give speeches while wearing similar clothing, with minor scene changes. Different actors were required to shoot in designated scenes such as offices, studies, and bedrooms. We used three DeepFake generation methods, namely, DeepFaceLab [66, 67], Faceswap [68, 69], and Faceswap-GAN [70], to generate 187 forged videos. DFS consists of 133 training videos and 54 test videos, totaling 578,613 frames. DFS is an Asian face dataset, and all actors authorized the modification of their recorded videos.

5.2.3. Video Inpainting Dataset. Yu et al. [8] proposed a video inpainting dataset named DAVIS-VI based on DAVIS [71]. They used three video inpainting methods, namely, OPN [72], CPNET [73], and DVI [74], to remove the annotated objects from the DAVIS dataset and generate corresponding inpainted videos. However, due to the limited number of original samples, we further augmented the DAVIS-VI dataset with three additional video inpainting methods: FGVC [31], DFGVI [30], and STTN [75]. As shown in Figure 5(c), DAVIS-VI contains 50 original videos and 300 inpainted videos, totaling 33,550 frames. The training set includes 200 inpainted videos, and the test set includes 100 inpainted videos.

5.2.4. Video Splicing Dataset. Video splicing detection receives relatively less attention due to the lack of video splicing datasets. Compared to image splicing datasets, creating a video splicing dataset is challenging because it requires considering the position, size, color, and semantics of spliced objects. As shown in Figure 5(b), we create a video

(a) (b) (c)

Figure 5: Thumbnails of the datasets. (a) DFS: a dataset for detecting DeepFake of similar (DFS) scene videos. (b) Video splicing: a dataset for detecting video splicing. (c) DAVIS-VI: a dataset for detecting video inpainting based on DAVIS2016.

splicing dataset called video splicing to evaluate the performance of our method in detecting video splicing forgery. The video splicing dataset contains 30 carefully manually created videos of different scenes as the test set and 795 randomly spliced forgery videos based on these objects and real videos as the training set. We develop a Photoshop-like tool to create videos frame by frame. Given all the frames of a real video $\mathbb{A} = a_1, a_2, \ldots, a_n$, where n is the number of frames in the real video, and a set of frames for the object to be spliced $\mathbb{B} = b_1, b_2, \ldots, b_m$, where m is the number of frames in the object to be spliced and $m < n$, the frames of the synthesized video are defined as $\mathbb{R} = r_1, r_2, \ldots, r_m$. The production process of the forged video is defined as follows:

$$\mathbb{R} = \mathbb{A} + (\text{scale} \times \mathbb{B} + \text{pos}), \tag{9}$$

where scale is the scaling factor of the spliced object and pos is the position of the object $\mathbb{B}$ in the forged video $\mathbb{R}$.

5.3. Robustness Experiments. The propagation of fake videos on the Internet inevitably involves various video processing techniques, such as compression, cropping, redrawing, and blurring. Improving the robustness of video detection against these operations has important practical significance. As shown in Table 1, the performance of processed videos is almost the same as that of unprocessed videos. This is mainly because video processing techniques destroy the forgery traces of fake videos, and our method extracts features that are irrelevant to forgery, thus having better robustness. In addition, longer hash codes usually lead to better performance. However, a slight performance decrease is observed when the hash code length reaches 1,024. More hash code elements can better capture small differences between different videos, help generate better hash centers, and solve the nondifferentiable optimization problem of similar videos due to the marginal utility. However, when the number of hash code elements is too high, the marginal utility decreases. Therefore, when the hash code length exceeds 512, redundant information is learned which may have a negative impact on source tracing. The experiments prove the robustness of our method against video processing on Internet scenes.

5.4. Evaluations of Cross-Dataset. We conduct cross-dataset evaluations to further validate the generalization ability of our proposed method. As shown in Table 2, our method achieves comparable or better performance on within-dataset compared to recent works but has a significant advantage on cross-dataset. This is because those methods simply learn dataset-dependent forgery features from existing data, which may not be applicable to unknown forgery data. However, our method aims to learn more general features that are independent of forgery methods. Experiments show that our method has better generalization ability for detecting unknown forgeries.

5.5. DeepFake Comparison Experiment. To evaluate the performance of our method, we compare it with the state-of-the-art methods on popular datasets including Celeb-DF, DeepFakeDetection, and FaceForensics++. Figure 6(a) presents several correct result examples on the FaceForensics++ dataset, where "Fake" indicates forged videos and "Traced" represents the traced videos. As shown in Table 3, our method achieves comparable or better performance than the state-of-the-art methods. As shown in Table 4, our method performs consistently well on different qualities and types of DeepFake videos, achieving better performance than existing methods, especially on low-quality (LQ) videos. Existing methods rely on learning forgery features from the data, which results in good performance on the same dataset. However, the reason for the poor performance on the LQ dataset is that LQ videos damage the potential forgery features they learned. In contrast, our method extracts features that are independent of forgery traces, resulting in better performance in detecting various types of forgeries and low-quality videos. The experiment shows that our method is effective in detecting DeepFake videos on multiple datasets and is more robust than existing methods.

5.6. Experiment on Video Inpainting Detection. To verify the performance of our method in detecting video object removal, we are conducting experiments on the DAVIS-VI

Table 1: Robustness experiments with various video preprocessing and different hash bits on FaceForensics++.

Video processing	FF++ raw (bits)					FF++ C23 (bits)					FF++ C40 (bits)				
	64	128	256	512	1024	64	128	256	512	1024	64	128	256	512	1024
None	**0.852**	0.932	0.948	0.998	**0.991**	**0.847**	0.944	0.944	0.998	0.990	0.846	0.941	**0.946**	0.997	0.991
Sharpening	0.850	0.930	**0.949**	**0.999**	**0.991**	0.845	0.943	**0.945**	**0.999**	0.989	0.847	0.942	0.945	0.996	0.991
Noise	0.844	**0.937**	0.944	**0.999**	0.990	0.844	0.933	0.940	**0.999**	**0.991**	**0.853**	**0.944**	0.942	**0.999**	0.991
Blur	0.846	0.934	0.947	0.998	**0.991**	0.844	0.944	0.939	**0.999**	**0.991**	0.848	0.942	0.941	**0.999**	0.991
Median filter	0.850	0.935	0.948	0.998	**0.991**	0.844	**0.945**	0.941	0.998	**0.991**	0.851	0.942	**0.946**	0.997	**0.992**
Video crop	0.633	0.801	0.862	0.983	0.963	0.636	0.862	0.814	0.986	0.962	0.629	0.816	0.859	0.988	0.964

Bold values represent the best results in the correlation domain.

Table 2: Evaluation of cross-dataset performance on five subsets of FaceForensics++ (FF++): DeepFake (DF), Face2Face (F2F), Faceswap (FS), NeuralTexture (NT), and FaceShifter (FSh).

Training set	Methods	Test set (ACC)				
		DF	F2F	FS	NT	FS
DF	Xception [54]	**99.3**	73.6	49.0	73.6	—
	HRNet [51]	**99.3**	68.2	39.1	71.4	—
	Face X-ray [52]	98.7	63.3	60.0	69.8	—
	ADD [1]	98.7	—	—	—	—
	Grad-CAM [53]	99.2	0.76.4	49.7	81.4	—
	Ours	98.8	**0.98.8**	**98.8**	**99.1**	**98.6**
F2F	Xception [54]	80.3	99.4	76.2	69.6	—
	HRNet [51]	83.6	**99.5**	56.6	61.3	—
	Face X-ray [52]	63.0	98.4	93.8	94.5	—
	ADD [1]	—	96.8	—	—	—
	Grad-CAM [53]	83.7	99.4	98.7	98.4	—
	Ours	**99.2**	99.4	**99.2**	**99.2**	**99.2**
FS	Xception [54]	66.4	88.8	99.4	71.3	—
	HRNet [51]	63.6	64.1	99.2	68.9	—
	Face X-ray [52]	45.8	96.1	98.1	95.7	—
	ADD [1]	—	—	97.9	—	—
	Grad-CAM [53]	68.5	99.3	99.5	98.0	—
	Ours	**99.9**	**99.8**	**99.9**	**99.8**	**99.9**
NT	Xception [54]	79.9	81.3	73.1	99.1	—
	HRNet [51]	94.1	87.3	64.1	98.6	—
	Face X-ray [52]	70.5	91.7	91.0	92.5	—
	ADD [1]	—	—	—	88.5	—
	Grad-CAM [53]	89.4	**99.5**	**99.3**	**99.4**	—
	Ours	**99.3**	99.2	**99.3**	99.3	**99.3**
FS	ADD [1]	—	—	—	—	96.6
	Ours	**98.8**	**98.8**	**99.0**	**99.3**	**99.1**

We trained on one subset and tested on the other four subsets. Bold values represent the best results in the correlation domain.

dataset [8]. Existing video object removal detection methods suffer from pixel-level detection and lack corresponding comparison methods. Figure 6(b) presents several examples of correct results on the DAVIS-VI dataset, where "Fake" denotes forged videos and "Traced" refers to traced videos. As shown in Table 5, our method achieves nearly 100% accuracy. This is because the object removal dataset contains only 50 real videos, making it easy to find the original videos from the 50 real videos. Due to the large semantic differences between the forged videos in video object removal and the real videos, it is easier to learn the differences between videos and make the source-tracing task relatively simple. The experiments are showing that our method is effective in detecting video inpainting.

5.7. Experiment on Video Splicing Detection. To evaluate the performance of our proposed method for detecting video splicing, we conducted experiments on the video splicing dataset. However, due to the lack of publicly available datasets for video splicing, there are no comparable methods. Figure 6(c) presents several examples of correct results on the video splicing dataset, where "Fake" denotes forged videos and "Traced" refers to traced videos. As shown in Table 5, our method achieved nearly 100% accuracy in the experiment, which is mainly due to the small size of the test set consisting of only 30 videos. In addition, spliced videos often have significant semantic differences, making them easier to trace. These results demonstrate the effectiveness of our proposed method for detecting video splicing.

Figure 6: Experimental results on three different types of datasets are presented: DeepFake, video inpainting, and video splicing. (a) FF++, (b) DAVIS-VI, and (c) video splicing.

Table 3: Comparison of accuracy (ACC) with existing methods on multiple datasets.

Methods	Celeb-DF	DFD	FF++ (HQ)
Xception [54]	0.994	0.831	0.994
Face X-ray [52]	**0.996**	0.856	0.960
Grad-CAM [53]	0.794	0.919	0.992
STIL [55]	0.996	—	0.986
ISTVT [3]	0.998	—	0.996
MRL [56]	**0.999**	—	0.938
Ours	0.994	**0.963**	**0.999**

Bold values represent the best results in the correlation dataset.

Table 4: Comparison experiment of fine-grained accuracy (ACC) with recent works on FaceForensics++ high-quality (HQ) and low-quality (LQ) datasets.

Methods	FF++ (HQ)				FF++ (LQ)				Celeb-DF
	DF	F2F	FS	NT	DF	F2F	FS	NT	
Xception	98.9	98.9	99.6	95.0	96.8	91.1	94.6	87.1	99.4
I3D	92.9	92.9	96.4	90.4	91.1	86.4	91.4	78.6	99.2
LSTM	99.6	99.3	98.2	93.9	96.4	88.2	94.3	88.2	95.7
TEI	97.9	97.1	97.5	94.3	95.0	91.1	94.6	90.4	99.1
ADDNet-3d	92.1	83.9	92.5	78.2	90.4	78.2	80.0	69.3	95.2
S-MIL	98.6	99.3	99.3	95.7	96.8	91.4	94.6	88.6	99.2
S-MIL-T	99.6	99.6	**100.0**	94.3	97.1	91.1	96.1	86.8	98.8
STIL	99.6	99.3	**100.0**	95.4	98.2	92.1	97.1	91.8	**99.8**
VTN	99.6	99.3	99.6	95.4	97.9	92.1	95.7	90.4	99.3
ISTVT	99.6	99.6	**100.0**	96.8	98.9	96.1	97.5	92.1	**99.8**
Ours	**99.9**	**99.9**	99.9	**0.999**	**99.9**	**100.0**	**99.9**	**99.9**	99.4

Table 5: Evaluation of classification and localization on different types of datasets.

Datasets	Localization (mIoU)			Classification (%)	
	DMAC [62]	DMVN [63]	Ours	ACC	Total originals
DFD	—	—	0.726	94.9	363
DFS	—	—	0.880	100.0	133
Video splicing	0.828	0.751	**0.842**	100.0	30
DAVIS-VI	0.828	0.751	**0.882**	100.0	50

Bold values represent the best results.

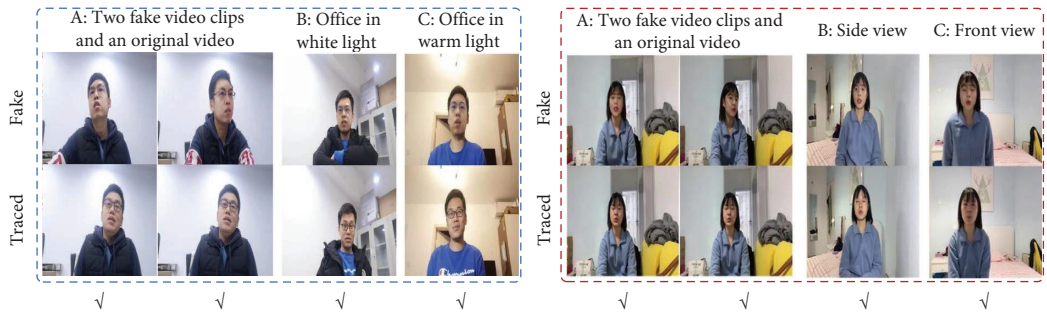

Figure 7: Example results of experiments on similar videos in two different scenarios: one with a similar background and another with the same background from a different angle.

5.8. Experiment on Similar Scene Detection. To evaluate the performance of our designed hash triplet loss in distinguishing similar videos on the DFS dataset, as shown in Table 5, we evaluated our method on 133 forged videos traced back to 54 real videos, achieving 100% accuracy. The large amount of data in the DFS dataset, which contains 578,613 frames, allows our method to fully exploit each video's unique features and exhibit excellent performance. We also analyzed the experimental results on similar videos. Figure 7 shows male and female subclips with similar backgrounds captured from different angles in the same room. Despite their similarity, our method can accurately identify the original video. The results demonstrate that the hash triplet loss can effectively learn subtle differences in similar videos and address the nondifferentiable optimization problem in hash code learning.

5.9. Localizator Evaluation Experiment. As shown in Table 5, our method outperforms DMAC and DMVN in localizing the suspicious regions of the two videos. Since these two methods are relatively early, we applied an effective feature extraction network based on ViT and CNN to more easily mark the differential regions of the two videos. The experiment shows that the localizator is effective in distinguishing the suspicious regions of the two videos.

5.10. Ablation Study. To validate the effectiveness of the hash triplet loss, we evaluated our method from three aspects: structure, activation function, and error analysis. Since the average Hamming distance has a significant impact on the quality of the generated hash centers, we used it as one of the evaluation metrics [13].

5.10.1. Hash Triplet Loss. To validate the effectiveness of the hash triplet loss structure, we evaluated the performance using only the interclass or intraclass loss in FaceForensics++. As shown in Figure 8(a), when trained with only the interclass loss, it is difficult to train the intraclass videos to be similar to the hash center. Thus, the hash center keeps changing, which cannot meet our expectations. When trained with only the intraclass loss, despite various improved algorithms and different training strategies that we have attempted, the hash is always unstable and close to $\vec{0}$ or $\vec{1}$. When both losses are trained together, the average Hamming distance of the hash center gradually approaches half of the hash bits. The experimental results demonstrate that the structure of the hash triplet loss is reasonable and necessary.

5.10.2. Various Activation Functions

$$\mathscr{F}_{\tanh} = \frac{(\operatorname{sign}(x)+1)}{2} \in \{0,1\}^k, \tag{10}$$

$$\mathscr{F}_{\text{sigmoid}} = \frac{(\operatorname{sign}(x-0.5)+1)}{2} \in \{0,1\}^k, \tag{11}$$

$$\mathscr{F}_{\text{relu}} = \operatorname{sign}(x) \in \{0,1\}^k. \tag{12}$$

We evaluated the performance of different activation functions and their corresponding hash binary functions, such as ReLU with equation (10), tahn with equation (11), and sigmoid with equation (12). As shown in Figure 8(b), with the help of hash triplet loss, the Hamming distance of these activation functions can quickly stabilize at around half of the hash bits. This suggests that the influence of different activation functions on the experimental results is minor, while hash triplet loss is more important to the experimental results.

5.10.3. Analysis of Incorrect Results. As shown in Figure 9, we present examples of erroneous results on multiple datasets. The first three videos share similar backgrounds and human poses, except for differences in the faces and clothing. In the fourth video, two people swapped positions. The remaining videos have subtle differences that are even imperceptible to human observers. These errors are reasonable and consistent with common sense. In our extended experiments, we found that expanding the scope of tracing (Top-10) can avoid these errors. The errors in these experimental results indicate that our method is sensitive to the structural content of videos. This demonstrates that our method effectively learns the semantic structure of the video, rather than relying on forgery traces. This property is beneficial for improving the detection of unknown forgery videos.

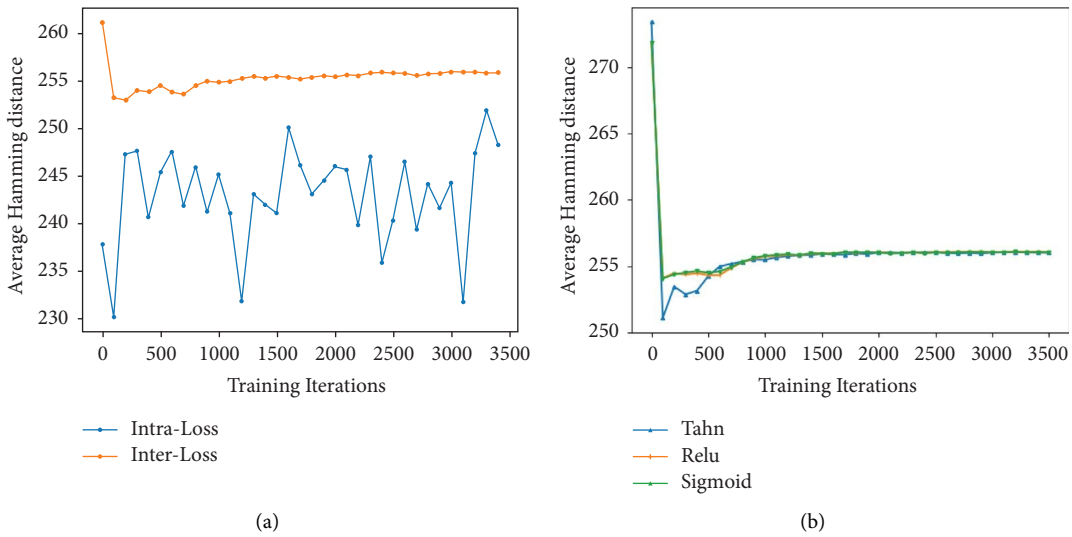

FIGURE 8: Ablation studies of the hash triplet loss: average Hamming distance with various activation functions and different losses. (a) Hash losses and (b) activation functions.

FIGURE 9: We evaluate multiple datasets to analyze incorrect tracing results, which help us understand the focus of our method and analyze whether the features learned by our method are effective in improving its performance. We believe that these errors are reasonable and can be avoided by some software design techniques.

6. Conclusions

In this paper, we propose a reliable source-tracing-based method for detecting forged videos, which provides trustworthy and interpretable detection results. Our method is essential for scenarios that require reliable detection to prevent the spread of rumors on the Internet. We introduce the hash triplet loss to solve the nondifferentiable optimization problem of similar videos, which effectively improves source tracing accuracy and the ability to distinguish similar videos. Experimental results on various types of datasets demonstrate that our method is capable of detecting video forgeries and exhibits good robustness to various commonly used video processing techniques on the Internet. Since our method extracts forgery-independent features, it can be easily extended to detect other types of video synthesis forgeries. In conclusion, our proposed method provides an efficient and reliable solution for detecting forged videos and has great potential for industrial applications in the future.

Data Availability

The dataset used in this paper is publicly available on the internet: DFS (https://pan.baidu.com/s/1rBB_znROfLIXTTiaPrP5Ng?pwd=DFS0), DAVIS-VI (https://pan.baidu.com/s/1kLi_JZygE_JDkY7HYt8Oyg?pwd=VIN0), and VideoSplicing (https://pan.baidu.com/s/10SBHYpN3nB3pkJH3_IBnHg?pwd=VS00).

Consent

The images collected as part of the DFS datasets were collected with consent to publish from the participants, with the understanding that this may be used for future research purposes.

Disclosure

A preprint of our manuscript was published in https://arxiv.org/abs/2112.08117 [76].

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Key Technology Research and Development Program under 2020AAA0140000.

References

[1] L. M. Binh and S. S. Woo, "ADD: frequency attention and multi-view based knowledge distillation to detect low-quality compressed deepfake images," in *AAAI*, pp. 122–130, AAAI Press, Washington, DC, USA, 2022.

[2] H. Wu, J. Zhou, J. Tian, J. Liu, and Y. Qiao, "Robust image forgery detection against transmission over online social networks," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 443–456, 2022.

[3] C. Zhao, C. Wang, G. Hu, H. Chen, C. Liu, and J. Tang, "ISTVT: interpretable spatial-temporal video transformer for deepfake detection," *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 1335–1348, 2023.

[4] Z. Li, C. Lu, J. Qin, C. Guo, and M. Cheng, "Towards an end-to-end framework for flow-guided video inpainting," in *Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17541–17550, IEEE, New Orleans, LA, USA, June, 2022.

[5] J. Ren, Q. Zheng, Y. Zhao, X. Xu, and C. Li, "Dlformer: discrete latent transformer for video inpainting," in *Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3501–3510, IEEE, New Orleans, LA, USA, June, 2022.

[6] C. Dong, X. Chen, R. Hu, J. Cao, and X. Li, "Mvss-net: multi-view multi-scale supervised networks for image manipulation detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 3, pp. 3539–3553, 2023.

[7] Z. Gu, Y. Chen, T. Yao, S. Ding, J. Li, and L. Ma, "Delving into the local: dynamic inconsistency learning for deepfake video detection," in *Proceedings of the AAAI*, pp. 744–752, AAAI Press, Washington, DC, USA, August, 2022.

[8] B. Yu, W. Li, X. Li, J. Lu, and J. Zhou, "Frequency-aware spatiotemporal transformers for video inpainting detection," in *Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 8188–8197, Montreal, QC, Canada, October, 2021.

[9] P. Zhou, N. Yu, Z. Wu, L. Davis, A. Shrivastava, and S. Lim, "Deep video inpainting detection," in *Proceedings of the 32nd British Machine Vision Conference 2021, BMVC*, p. 35, London, UK, November, 2021.

[10] L. Yuan, T. Wang, X. Zhang et al., "Central similarity quantization for efficient image and video retrieval," in *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3080–3089, IEEE, Seattle, WA, USA, June, 2020.

[11] X. Xiang, Y. Zhang, L. Jin, Z. Li, and J. Tang, "Sub-region localized hashing for fine-grained image retrieval," *IEEE Transactions on Image Processing*, vol. 31, pp. 314–326, 2022.

[12] Y. Liang, Y. Pan, H. Lai, W. Liu, and J. Yin, "Deep listwise triplet hashing for fine-grained image retrieval," *IEEE Transactions on Image Processing*, vol. 31, pp. 949–961, 2022.

[13] Y. Shen, J. Qin, J. Chen et al., "Auto-encoding twin-bottleneck hashing," in *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2815–2824, IEEE, Seattle, WA, USA, June, 2020.

[14] W. Wang, E. Xie, X. Li et al., "Pvtv2: improved baselines with pyramid vision transformer," *Computational Visual Media*, vol. 8, p. 13797, 2021.

[15] A. Arnab, M. Dehghani, G. Heigold, C. Sun, M. Lučić, and C. Schmid, "Vivit: a video vision transformer," in *Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6816–6826, Montreal, QC, Canada, June, 2021.

[16] Y. Hu, H. Zhao, Z. Yu, B. Liu, and X. Yu, "Exposing deepfake videos with spatial, frequency and multi-scale temporal artifacts," in *Digital Forensics and Watermarking - 20th International Workshop, IWDW 2021*, X. Zhao, A. Piva, and P. C. Alfaro, Eds., vol. 13180, pp. 47–57, Springer, Beijing, China, 2021.

[17] 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. 6458–6467, IEEE, Nashville, TN, USA, June, 2021.

[18] 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, IEEE, Nashville, TN, USA, June, 2021.

[19] Z. Sun, Y. Han, Z. Hua, N. Ruan, and W. Jia, "Improving the efficiency and robustness of deepfakes detection through precise geometric features," in *Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3609–3618, IEEE, Nashville, TN, USA, June, 2021.

[20] D. Cozzolino, A. Rössler, J. Thies, M. Nießner, and L. Verdoliva, "Id-reveal: identity-aware deepfake video detection," in *Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15088–15097, Montreal, QC, Canada, March, 2021.

[21] 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.

[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, IEEE, Nashville, TN, USA, June, 2021.

[23] P. Pei, X. Zhao, Y. Cao, and C. Hu, "Visual explanations for exposing potential inconsistency of deepfakes," in *Digital Forensics and Watermarking - 21st International Workshop, IWDW 2022*, X. Zhao, Z. Tang, P. C. Alfaro, and A. Piva, Eds., vol. 13825, pp. 68–82, Springer, Guilin, China, 2022.

[24] M. Li, Y. Ahmadiadli, and X. Zhang, "A comparative study on physical and perceptual features for deepfake audio detection," in *Proceedings of the DDAM@MM 2022: Proceedings of the 1st International Workshop on Deepfake Detection for Audio Multimedia*, J. Tao, H. Li, H. Meng et al., Eds., pp. 35–41, ACM, Lisboa, Portugal, October 2022.

[25] J. Xue, C. Fan, Z. Lv et al., "Audio deepfake detection based on a combination of F0 information and real plus imaginary spectrogram features," in *Proceedings of the DDAM@MM 2022: Proceedings of the 1st International Workshop on Deepfake Detection for Audio Multimedia*, J. Tao, H. Li,

H. Meng et al., Eds., pp. 19–26, Lisboa, Portugal, October, 2022.

[26] Y. Huang, F. Juefei-Xu, Q. Guo, Y. Liu, and G. Pu, "Fakelocator: robust localization of gan-based face manipulations," *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 2657–2672, 2022.

[27] M. Ebdelli, O. L. Meur, and C. Guillemot, "Video inpainting with short-term windows: application to object removal and error concealment," *IEEE Transactions on Image Processing*, vol. 24, no. 10, pp. 3034–3047, 2015.

[28] Z. Li, C.-Z. Lu, J. Qin, C.-L. Guo, and M.-M. Cheng, "Towards an end-to-end framework for flow-guided video inpainting," in *Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, New Orleans, LA USA, June, 2022.

[29] R. Szeto and J. J. Corso, "The DEVIL is in the details: a diagnostic evaluation benchmark for video inpainting," in *Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, New Orleans, LA, USA, June, 2022.

[30] R. Xu, X. Li, B. Zhou, and C. C. Loy, "Deep flow-guided video inpainting," in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3723–3732, IEEE, Long Beach, CA, USA, June, 2019.

[31] C. Gao, A. Saraf, J. Huang, and J. Kopf, "Flow-edge guided video completion," in *Proceedings of the European Conference on Computer Vision*, vol. 12357, pp. 713–729, Glasgow, UK, September, 2020.

[32] X. Zhu, Y. Qian, X. Zhao, B. Sun, and Y. Sun, "A deep learning approach to patch-based image inpainting forensics," *Signal Processing: Image Communication*, vol. 67, pp. 90–99, 2018.

[33] A. Li, Q. Ke, X. Ma et al., "Noise doesn't lie: towards universal detection of deep inpainting," in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, vol. 8, pp. 786–792, Montreal, Canada, August, 2021.

[34] H. Li and J. Huang, "Localization of deep inpainting using high-pass fully convolutional network," in *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 8300–8309, Seoul, Korea (South), October, 2019.

[35] Y. Niu, B. Tondi, Y. Zhao, R. Ni, and M. Barni, "Image splicing detection," *Localization and Attribution Via JPEG Primary Quantization Matrix Estimation and Clustering*, vol. 16, pp. 5397–5412, 2021.

[36] Y. Rao and J. Ni, "Self-supervised domain adaptation for forgery localization of JPEG compressed images," in *Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15014–15023, Montreal, QC, Canada, October, 2021.

[37] X. Xiang, Y. Zhang, L. Jin, Z. Li, and J. Tang, "Sub-region localized hashing for fine-grained image retrieval," *IEEE Transactions on Image Processing*, vol. 31, pp. 314–326, 2022.

[38] L. Xu, X. Zeng, W. Li, and L. Bai, "Idhashgan: deep hashing with generative adversarial nets for incomplete data retrieval," *IEEE Transactions on Multimedia*, vol. 24, pp. 534–545, 2022.

[39] W. Shang, M. Liu, W. Lin, and M. Jia, "Tracing the source of news based on blockchain," in *Proceedings of the 2018 IEEE/ACIS 17th International Conference on Computer and Information Science (ICIS)*, pp. 377–381, Singapore, June, 2018.

[40] A. D. Dwivedi, R. Singh, S. Dhall, G. Srivastava, and S. K. Pal, "Tracing the source of fake news using a scalable blockchain distributed network," in *Proceedings of the 2020 IEEE 17th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, pp. 38–43, Delhi, India, December, 2020.

[41] G. Shrivastava, P. Kumar, R. P. Ojha, P. K. Srivastava, S. Mohan, and G. Srivastava, "Defensive modeling of fake news through online social networks," *IEEE Transactions on Computational Social Systems*, vol. 7, no. 5, pp. 1159–1167, 2020.

[42] T. Zhu, X. Wang, X. Qin, and M. Li, "Source tracing: detecting voice spoofing," *CoRR*, Article ID 08601, 2022.

[43] Y. Yuan, R. Fu, L. Huang et al., "Hrformer: high-resolution vision transformer for dense predict," in *Proceedings of the Conference on Neural Information Processing Systems*, pp. 7281–7293, Montreal, Canada, December, 2021.

[44] Q. Diao, Y. Jiang, B. Wen, J. Sun, and Z. Yuan, "Metaformer: a unified meta framework for fine-grained recognition," in *Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, New Orleans, LA, USA, June, 2022.

[45] Y. Jiang, S. Chang, and Z. Wang, "Transgan: two pure transformers can make one strong gan, and that can scale up," in *Proceedings of the Conference on Neural Information Processing Systems*, pp. 14745–14758, Montreal, Canada, December, 2021.

[46] R. Liu, H. Deng, Y. Huang et al., "Fuseformer: fusing fine-grained information in transformers for video inpainting," in *Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14020–14029, Montreal, QC, Canada, June, 2021.

[47] A. Vaswani, N. Shazeer, N. Parmar et al., "Attention is all you need," in *Proceedings of the Conference on Neural Information Processing Systems*, pp. 5998–6008, Long Beach, CA, USA, March, 2017.

[48] W. Wang, E. Xie, X. Li et al., "Pyramid vision transformer: a versatile backbone for dense prediction without convolutions," in *Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 548–558, Montreal, QC, Canada, June, 2021.

[49] J. Guo, K. Han, H. Wu et al., "Cmt: convolutional neural networks meet vision transformers," in *Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, New Orleans, LA, USA, June, 2022.

[50] 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 (South), October, 2019.

[51] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5693–5703, IEEE, Long Beach, CA, USA, June, 2019.

[52] L. Li, J. Bao, T. Zhang et al., "Face x-ray for more general face forgery detection," in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5000–5009, Seattle, WA, USA, June, 2020.

[53] 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. 16317–16326, IEEE, Nashville, TN, USA, June, 2021.

[54] F. Chollet, "Xception: deep learning with depthwise separable convolutions," in *Proceedings of the 2017 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1800–1807, IEEE, Honolulu, HI, USA, August, 2017.

[55] Z. Gu, Y. Chen, T. Yao et al., "Spatiotemporal inconsistency learning for deepfake video detection," in *Proceedings of the 2021 19th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, H. T. Shen, Y. Zhuang, J. R. Smith et al., Eds., pp. 3473–3481, Stockholm, Sweden, September, 2021.

[56] Z. Yang, J. Liang, Y. Xu, X. Zhang, and R. He, "Masked relation learning for deepfake detection," *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 1696–1708, 2023.

[57] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *Proceedings of the 2017 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4724–4733, IEEE, Honolulu, HI, USA, June, 2017.

[58] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[59] Z. Liu, D. Luo, Y. Wang et al., "Teinet: towards an efficient architecture for video recognition," in *Proceedings of the The Thirty-Fourth AAAI Conference on Artificial Intelligence*, pp. 11669–11676, AAAI Press, New York, NY, USA, February, 2020.

[60] B. Zi, M. Chang, J. Chen, X. Ma, and Y.-G. Jiang, "Wilddeepfake: a challenging real-world dataset for deepfake detection," in *Proceedings of the MM '20: The 28th ACM International Conference on Multimedia*, pp. 2382–2390, New York, NY, USA, October, 2020.

[61] S. A. Khan and H. Dai, "Video transformer for deepfake detection with incremental learning," in *Proceedings of the MM '20: The 28th ACM International Conference on Multimedia*, pp. 1821–1828, New York, NY, USA, August, 2021.

[62] Y. Liu, X. Zhu, X. Zhao, and Y. Cao, "Adversarial learning for constrained image splicing detection and localization based on atrous convolution," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 10, pp. 2551–2566, 2019.

[63] Y. Wu, W. Abd-Almageed, and P. Natarajan, "Deep matching and validation network: an end-to-end solution to constrained image splicing localization and detection," in *Proceedings of the MM '17: Proceedings of the 25th ACM international conference on Multimedia*, pp. 1480–1502, Mountain View, CA, USA, October, 2017.

[64] N. Dufour and A. Gully, "Deepfakedetection dataset," 2019, https://ai.googleblog.com/2019/09/contributing-data-to-deepfake-detection.html.

[65] 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, IEEE, Seattle, WA, USA, June, 2020.

[66] I. Perov, D. Gao, N. Chervoniy et al., "Deepfacelab: a simple, flexible and extensible face swapping framework," *CoRR*, Article ID 05535, 2020.

[67] iperov and L. Barr, "Tools: Deepfacelab," 2018, https://github.com/iperov/DeepFaceLab.

[68] GitHub, "Faceswap," 2018, https://github.com/MarekKowalski/FaceSwap/.

[69] I. Korshunova, W. Shi, J. Dambre, and L. Theis, "Fast face-swap using convolutional neural networks," in *Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 3697–3705, Venice, Italy, June, 2017.

[70] N. V.D. S. shaoanlu and C. Leung, "tools: faceswap-gan," 2018, https://github.com/shaoanlu/faceswap-GAN.

[71] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. V. Gool, M. H. Gross, and A. Sorkine-Hornung, "A benchmark dataset and evaluation methodology for video object segmentation," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 724–732, IEEE, Las Vegas, NV, USA, June, 2016.

[72] S. W. Oh, S. Lee, J. Lee, and S. J. Kim, "Onion-peel networks for deep video completion," in *Proceedings of the 2019 IEEE International Conference on Computer Vision (ICCV)*, pp. 4402–4411, Seoul, Korea (South), October, 2019.

[73] S. Lee, S. W. Oh, D. Won, and S. J. Kim, "Copy-and-paste networks for deep video inpainting," in *Proceedings of the 2019 IEEE International Conference on Computer Vision (ICCV)*, pp. 4412–4420, Seoul, Korea (South), October, 2019.

[74] D. Kim, S. Woo, J. Lee, and I. S. Kweon, "Deep video inpainting," in *Proceedings of the 2019 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5792–5801, IEEE, Long Beach, CA, USA, June, 2019.

[75] Y. Zeng, J. Fu, and H. Chao, "Learning joint spatial-temporal transformations for video inpainting," in *Proceedings of the European Conference on Computer Vision*, vol. 12361, pp. 528–543, Glasgow, UK, June, 2020.

[76] P. Pei, X. Zhao, J. Li, Y. Cao, and X. Yi, "Vision transformer based video hashing retrieval for tracing the source of fake videos," *CoRR*, Article ID 08117, 2021.