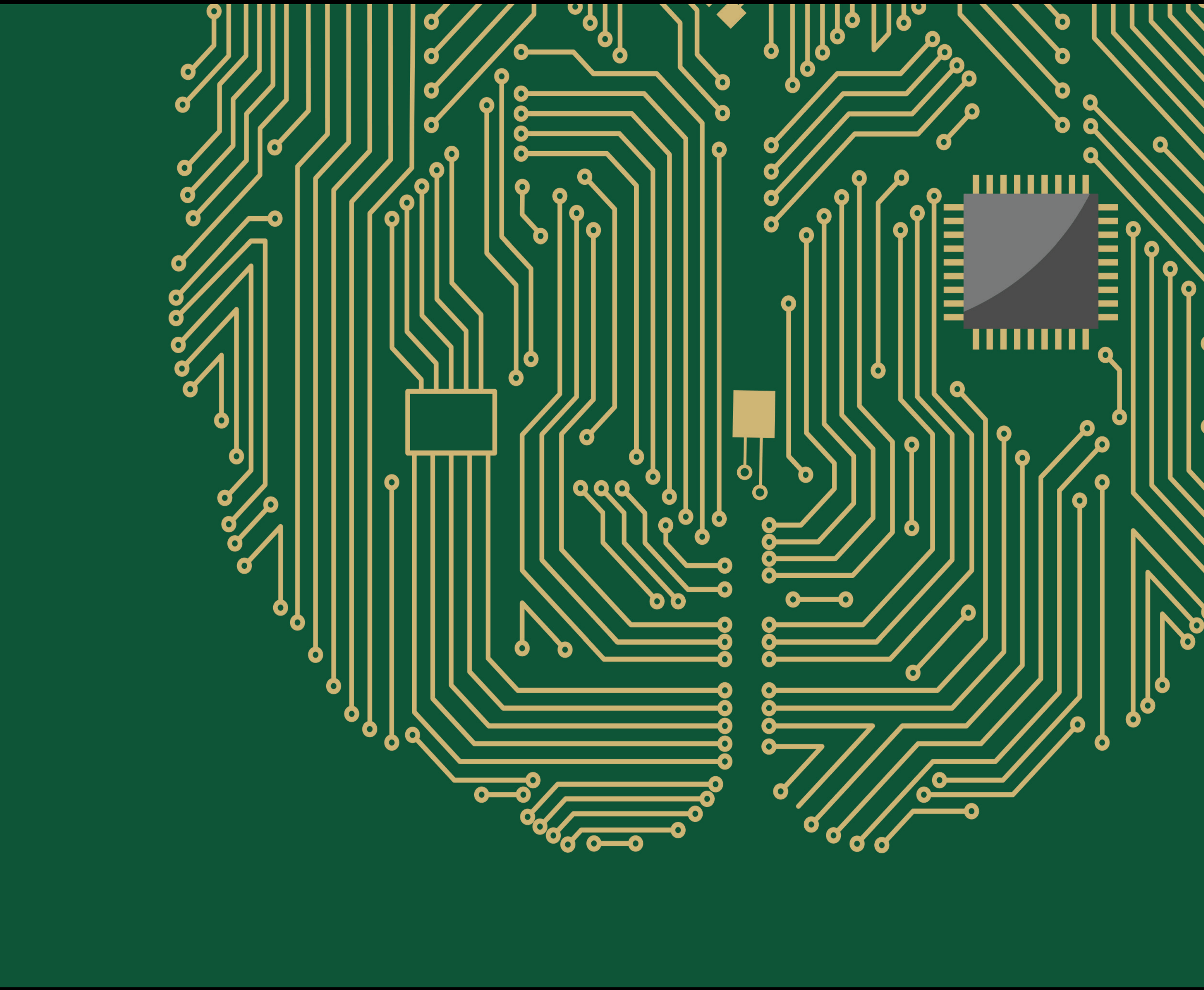
Computational Intelligence and Neuroscience

Deep Learning for Intelligent Surveillance Systems

Lead Guest Editor: Miguel Cazorla
Guest Editors: Francisco Gomez-Donoso, Jose Carlos Rangel, and Sergio Orts Escolano



Deep Learning for Intelligent Surveillance Systems

Computational Intelligence and Neuroscience

Deep Learning for Intelligent Surveillance Systems

Lead Guest Editor: Miguel Cazorla
Guest Editors: Francisco Gomez-Donoso, Jose Carlos Rangel, and Sergio Orts Escolano

Contents

Privacy-Preserved In-Cabin Monitoring System for Autonomous Vehicles
Ashutosh Mishra, Jaekwang Cha, and Shiho Kim
Research Article (15 pages), Article ID 5389359, Volume 2022 (2022)

Robust Real-Time Traffic Surveillance with Deep Learning
Jessica Fernández, José M. Cañas, Vanessa Fernández, and Sergio Paniego
Research Article (18 pages), Article ID 4632353, Volume 2021 (2021)

Integrated Multiscale Appearance Features and Motion Information Prediction Network for Anomaly Detection
Ting Liu, Chengqing Zhang, and Liming Wang
Research Article (13 pages), Article ID 6789956, Volume 2021 (2021)

A Smart Surveillance System for Uncooperative Gait Recognition Using Cycle Consistent Generative Adversarial Networks (CCGANs)
Wafaa Adnan Alsaggaf, Irfan Mehmood, Enas Fawai Khairullah, Samar Alhuraiji, Maha Farouk S. Sabir, Ahmed S. Alghamdi, and Ahmed A. Abd El-Latif
Research Article (12 pages), Article ID 3110416, Volume 2021 (2021)

NSD-SSD: A Novel Real-Time Ship Detector Based on Convolutional Neural Network in Surveillance Video
Jiuwu Sun, Zhijing Xu, and Shanshan Liang
Research Article (16 pages), Article ID 7018035, Volume 2021 (2021)

Anomaly Detection in Videos Using Two-Stream Autoencoder with Post Hoc Interpretability
Jiangfan Feng, Yukun Liang, and Lin Li
Research Article (15 pages), Article ID 7367870, Volume 2021 (2021)

Scaling Human-Object Interaction Recognition in the Video through Zero-Shot Learning
Vali Ollah Maraghi and Karim Faez
Research Article (15 pages), Article ID 9922697, Volume 2021 (2021)

Accurate Multilevel Classification for Wildlife Images
Francisco Gomez-Donoso, Félix Escalona, Ferran Pérez-Esteve, and Miguel Cazorla
Research Article (11 pages), Article ID 6690590, Volume 2021 (2021)

Hindawi
Computational Intelligence and Neuroscience
Volume 2022, Article ID 5389359, 15 pages
https://doi.org/10.1155/2022/5389359

Research Article

Privacy-Preserved In-Cabin Monitoring System for Autonomous Vehicles

Ashutosh Mishra, Jaekwang Cha, and Shiho Kim

School of Integrated Technology, YICT, Yonsei University, Seoul, Republic of Korea

Correspondence should be addressed to Shiho Kim; shiho@yonsei.ac.kr

Received 18 August 2021; Revised 28 October 2021; Accepted 21 March 2022; Published 22 April 2022

Academic Editor: Francisco Gomez-Donoso

Fully autonomous vehicles (FAVs) lack monitoring inside the cabin. Therefore, an in-cabin monitoring system (IMS) is required for surveilling people causing irregular or abnormal situations. However, monitoring in the public domain allows disclosure of an individual's face, which goes against privacy preservation. Furthermore, there is a contrary demand for privacy in the IMS of AVs. Therefore, an intelligent IMS must simultaneously satisfy the contrary requirements of personal privacy protection and person identification during abnormal situations. In this study, we proposed a privacy-preserved IMS, which can reidentify anonymized virtual individual faces in an abnormal situation. This IMS includes a step for extracting facial features, which is accomplished by the edge device (onboard unit) of the AV. This device anonymizes an individual's facial identity before transmitting the video frames to a data server. We created different abnormal scenarios in the vehicle cabin. Further, we reidentified the involved person by using the anonymized virtual face and the reserved feature vectors extracted from the suspected individual. Overall, the proposed approach preserves personal privacy while maintaining security in surveillance systems, such as for in-cabin monitoring of FAVs.

1. Introduction

Intelligent monitoring and surveillance systems are widely used to ensure safety and security. Popular applications of monitoring in public are video surveillance cameras (closed-circuit television); monitoring in intelligent transportation systems, including in-cabin monitoring and road traffic monitoring; and video monitoring for data generation and navigational tasks around city centers, airports, and public roads [1]. Driving automation also requires public visual information for multiple tasks [2]. The Society of Automotive Engineers defined six levels of autonomy in driving automation in 2014 (from no automation (level 0) to full automation (level 5)) [2–4]. Level 4 autonomous vehicles (AVs) are highly automated and capable of performing all driving tasks under certain conditions without human intervention. However, the driver (human) may control such AVs as and when required. In particular, fully autonomous vehicles (FAVs) (level 5 AVs) have no drivers; all occupants are passengers only [3, 4]. Therefore, no one oversees such AVs. In addition, in public and shared vehicles (such as ridesharing, carsharing, and car-full services in AVs), the passengers do not know each other. Therefore, it is important to ensure the security and safety of all occupants sitting in the cabin of such AVs. Furthermore, the vehicle should be protected from any malicious behavior of the occupants and/or external threats. Therefore, FAVs essentially require a multipronged in-cabin monitoring task in real time [5]. However, many countries have imposed a ban or severe restrictions on facial recognition techniques to secure personal information [6–16]. There are legal and ethical issues that impose various restrictions on public monitoring and surveillance systems [16–19]. Furthermore, identification of the accused is also important in abnormal (irregular) situations. This study was motivated by the fact that facial monitoring is important for safety; however, it poses a threat to individual privacy. In this study, we focused on the following two problems associated with in-cabin monitoring systems (IMSes):

(i) Protection of facial privacy.

(ii) Evidence of the accused in abnormal situations.

Therefore, a robust solution is required to provide privacy-preserved monitoring in public [20]. Moreover, it should be capable of identifying the concerned person when required. Figure 1 shows the dilemma of intelligent monitoring systems.

As illustrated in the above figure, an anonymous face protects personal information during in-cabin monitoring of an FAV. However, in certain irregular situations, personal identity is required to identify the accused person. An example of an abnormal incident or irregular situation can be an occupant of the FAV acting violently or attempting vandalism against the other occupants or toward the FAV itself. In such cases, it is important to identify the concerned person. Furthermore, this is an abnormal situation; however, in-cabin monitoring with real faces is not a solution to this problem. The breach of facial information leads to multiple consequences, such as misuse of facial data and banking and financial fraud [1, 6, 7, 13, 14]. One of our motivations for this work was to provide an approach that can protect against such problems in public monitoring systems, particularly the IMS. In-cabin monitoring with facial anonymization has security issues, while those with facial identity have privacy issues. Therefore, it creates a contradiction between privacy and security.

1.1. In-Cabin Monitoring. In-cabin monitoring is important in level 4 and beyond AVs [5]. It provides safety and security to the occupants. Simultaneously, it provides safety to the vehicle itself in an irregular situation. Past research works include in-cabin monitoring in various situations [21]. In-cabin monitoring for violence detection inside a FAV was reviewed in [22]. Bell et al. performed in-cabin monitoring to detect harsh vehicle maneuvers and risky driving behaviors [23]. Szawarski et al. patented the idea of in-cabin monitoring for a monitoring vehicle seat, occupants inside a vehicle, and the orientation of both the occupants and the vehicle seat [24]. Safety and cleaning problems of in-cabin monitoring of a vehicle were presented in [25]. However, a monitoring system should protect against any breach of personal privacy (facial identity) with the simultaneous ability to identify an actual person in case of irregular situations.

1.2. Facial Privacy versus Facial Recognition in Monitoring Applications. Real-time monitoring is essential in multiple monitoring applications. However, privacy in the public domain is an important concern in real-time monitoring tasks [26–30]. Facial anonymization is a common practice for preserving personal privacy. Recently, generative adversarial network- (GAN-) based deep learning (DL) models have been widely used for face swapping and anonymization [31–34]. In our previous study [31], we demonstrated a robust approach to preserving the facial identity of the occupants in a FAV cabin. It incorporated the facial swapping and reenactment technique to maintain privacy in in-cabin monitoring. However, in ab abnormal situation, the anonymized face of the occupants made it difficult to identify the concerned person [20].

1.3. Our Key Research Highlights. In this study, we propose an intelligent IMS. It is an efficient approach for identifying a person, even with an anonymized face. This method resolves both privacy and security issues. Accordingly, we can identify the person who causes an irregular situation, even with their anonymized face. In this approach, we preserved the key facial information of the occupants and stored these identity features on the cloud. These key features help in recognition of the person involved in the irregular situation. The highlights of this study are as follows:

(i) The concept of having an appropriate source face for each target face enhances puppeteering and reenactment of facial emotion and behavior. It helps in event and behavior detection in intelligent monitoring and surveillance systems in the public domain.

(ii) The involvement of the two-dimensional (2D) landmark position in the reenactment generator and separate segmentations of face and hair in the segmentation generator with inpainting and blending generators enhances the facial anonymization and reenactment operations.

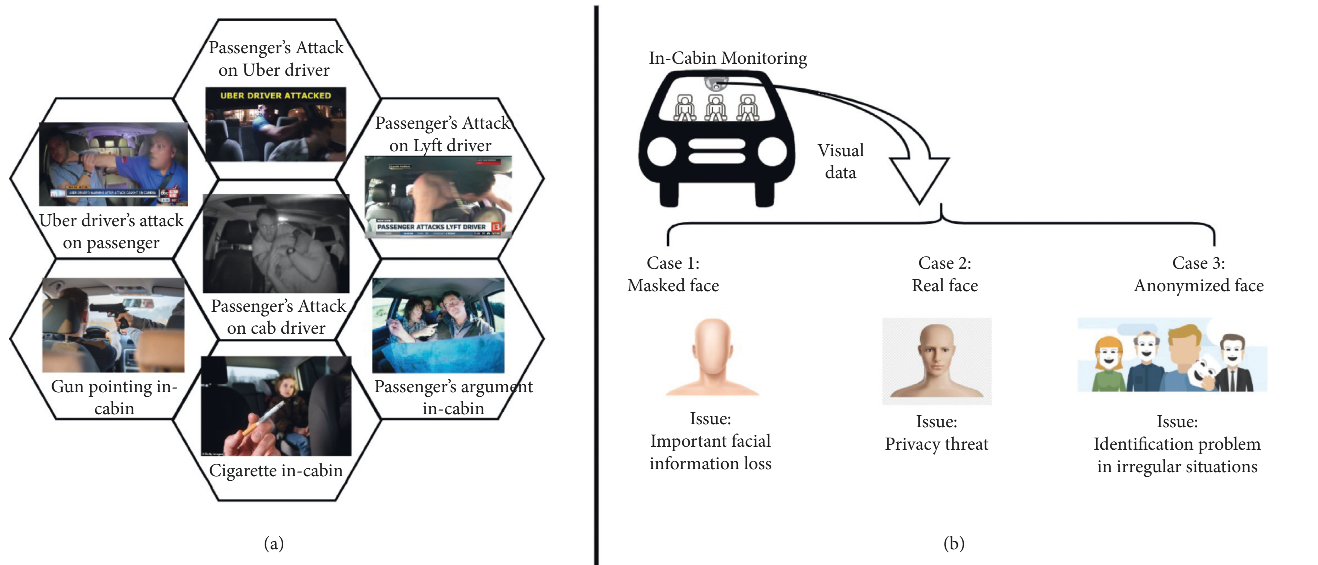
(iii) The 128D identity feature is a key marker for accurate facial identification in an anonymized domain. The concept of storing a pair of IDs (original and anonymized) leads to reidentification without any privacy threat. It is not possible to know the original face with only 128D identity features. For reidentification, both the original visual input and the ID are required. In the cloud, the anonymized visual image with the original *ID* is stored. Therefore, there is no threat of privacy breach, even though the *IDs* are stored in the cloud.

(iv) Therefore, the proposed approach augments the facial identity feature information to locate the involved person in any abnormal situation without any personal privacy breach.

This approach pioneers a newer method of monitoring and surveillance to avoid any legal or ethical issues. Therefore, a monitoring database can be created in the anonymized domain, thereby facilitating further research on events and behavior monitoring in the public domain.

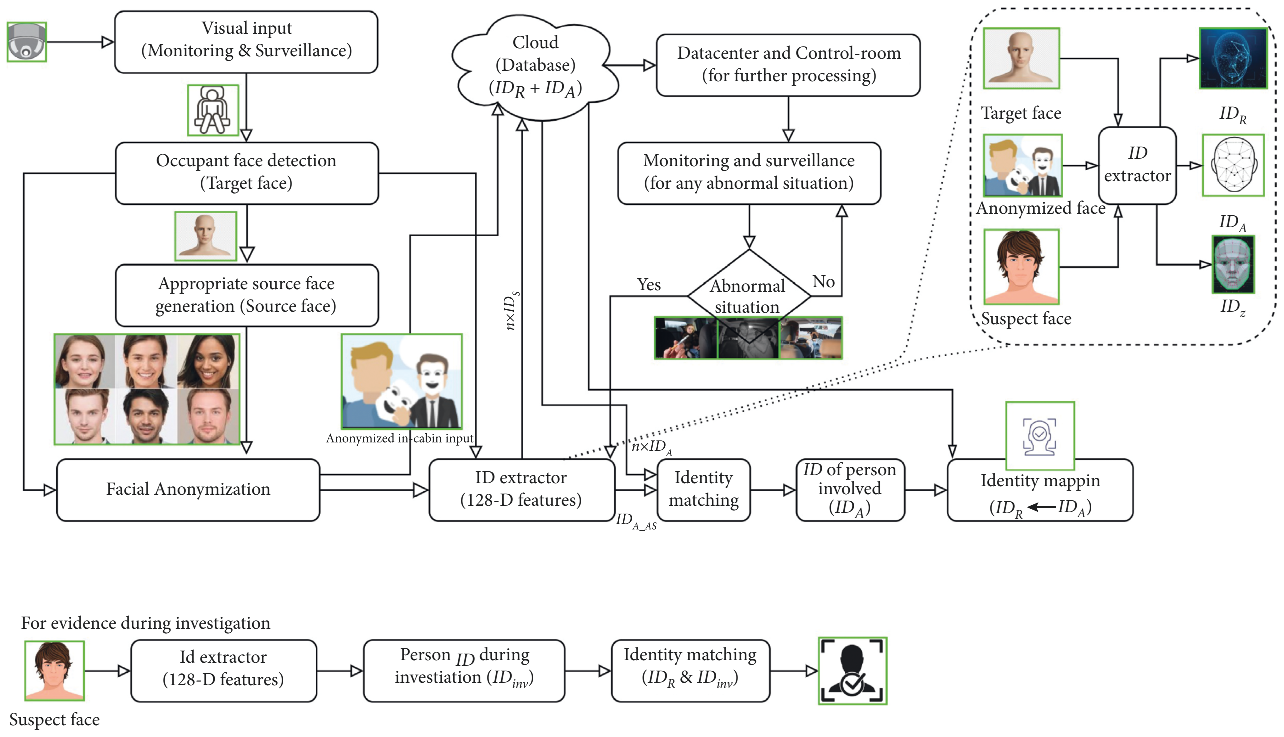
2. Materials and Methods

Personal privacy with identification is a challenge as well as a demand in real-time monitoring applications [20]. In this study, we developed a privacy-preserved IMS with the reidentification capability that can identify the accused person. The framework of the proposed method is shown in Figure 2. The proposed system operates in three stages. In stage 1, facial anonymization was performed to ensure personal privacy. It was performed using the onboard device of the



Passenger's Attack on Uber driver
UBER DRIVER ATTACKED
Passenger's Attack on Lyft driver
PASSENGER ATTACKS LYFT DRIVER
Uber driver's attack on passenger
Passenger's Attack on cab driver
Gun pointing in-cabin
Passenger's argument in-cabin
Cigarette in-cabin
(a)
In-Cabin Monitoring
Visual data
Case 1: Masked face
Case 2: Real face
Case 3: Anonymized face
Issue: Important facial information loss
Issue: Privacy threat
Issue: Identification problem in irregular situations
(b)

Figure 1: System overview of the proposed IMS. (a) Few examples causing abnormal situations in the cabin of a vehicle. (b) The dilemma of the legal and ethical issues (privacy) and practical problems (requirement of monitoring). Case 1: the masked face has no facial information, which is crucial in surveillance and monitoring inside the cabin of a vehicle. Case 2: real face suffers from personal privacy threats. Case 3: facial anonymization solves the problem of privacy; however, it has the problem of identifying the concerned person in case of irregular situations.



Visual input (Monitoring & Surveillance)
Occupant face detection (Target face)
Appropriate source face generation (Source face)
Facial Anonymization
Anonymized in-cabin input
Cloud (Database) ($ID_R + ID_A$)
$n \times ID_S$
ID extractor (128-D features)
ID_{A_AS}
$n \times ID_A$
Datacenter and Control-room (for further processing)
Monitoring and surveillance (for any abnormal situation)
Abnormal situation
Yes
No
Identity matching
ID of person involved (ID_A)
Identity mappin ($ID_R \leftarrow ID_A$)
Target face
Anonymized face
Suspect face
ID extractor
ID_R
ID_A
ID_z
For evidence during investigation
Suspect face
Id extractor (128-D features)
Person ID during investiation (ID_{inv})
Identity matching (ID_R & ID_{inv})

Figure 2: Proposed privacy-preserved intelligent IMS. Here, the identity features (*IDs*) are as follows: real face ID (ID_R), anonymized face ID (ID_A), ID of the occupant that caused an abnormal situation (ID_{A_AS}), and suspect face ID during the investigation (ID_{inv}).

AV. In stage 2, a pair of identity features (*IDs*) was generated for each face before and after anonymization (ID_R and ID_A). Further, the anonymized video along with the *IDs* was fed to the cloud. The pairs of *IDs* were kept in the cloud for person reidentification when required. The anonymized video frames were sent to the data center for further processing (monitoring and surveillance). In stage 3, the *IDs* were matched to search the accused (person involved in an irregular situation (ID_{A_AS})). During the investigation, the similarity between *ID*s ensured the identification of the concerned person (*ID*). Further, during the investigation, this approach was verified by matching the *IDs* of the suspect face (ID_{inv}) at the time of investigation with the accused person's *ID*.

The dilemma between monitoring requirements and legal and ethical issues is also resolved through this approach. The details of the proposed approach are discussed thoroughly in Section 2.2. This approach is suitable for creating a monitoring and surveillance database with legitimation.

2.1. Materials. Many research works have been published on personal privacy and person identification considering these two issues as separate research problems. In this study, we briefly surveyed the related works and developments on both face anonymization and person identification.

2.1.1. Face Anonymization. Face deidentification preserves privacy-sensitive information. It alters the original face to hide privacy-sensitive information. Anonymization of faces is an easier and more robust solution to personal privacy-related threats in the digital domain [35]. Blurring, masking faces, or creating a patch over faces is slightly easier than any other face anonymization approach; however, those methods suffer from significant loss of facial information [32, 36]. Therefore, face swapping has attracted significant attention for facial anonymization purposes. The morphable model-based facial exchange approach is considered a pioneering work in face swapping [37]. Bitouk et al. demonstrated automatic face replacement in their work [38]. Machine-learning-based face swapping was suggested in [39]. A convolutional neural network (CNN) was used for face segmentation and swapping in [40]. GAN-based deep models have become popular for virtual human face generation [33, 34]. Therefore, along with autoencoders, GAN-based face swapping has gained considerable attention among researchers for seamless end-to-end face anonymization [33, 34, 41]. Face swapping-based automatic generation and editing of faces was showcased in [42]. It used a region-separative GAN (RSGAN). An autoencoder-based algorithm for face swapping was presented to detect fake videos [43]. In [44], a GAN-based encoder-decoder network was suggested to swap human faces. Collateral privacy issues have also been resolved using the face swapping method [45]. Nirkin et al. suggested a face swapping GAN (FSGAN) in [46]. It provided subject agnostic face swapping and reenactment between a pair of faces. Naruniec et al. presented a fully automatic neural face swapping method in [47]. Sun et al. proposed a hybrid model for face anonymization [36]. Hukkelas et al. introduced a GAN-based DeepPrivacy architecture for face deidentification to remove all privacy-sensitive information [34].

2.1.2. Person Identification. Facial recognition has multipurpose objectives, such as recognition, classification, and discrimination. Urbanization and smart cities demand widespread applications for face recognition [48–52]. Therefore, various face recognition approaches involving person identification have been demonstrated by past researchers. Face recognition approaches are classified into three categories: local, holistic, and hybrid approaches [52]. Local approaches involve only partial facial features (such as eyes, mouth, and nose) to recognize a face, whereas holistic approaches involve complete facial features, including background for facial recognition. Hybrid approaches, as the name suggests, involve both local and holistic approaches. In holistic approaches, popular algorithms involve independent component analysis, linear discriminative analysis, and principal component analysis [53, 54]. The development of artificial intelligence (AI) incorporating DL and CNNs has boosted the performance of facial recognition algorithms. Taigman et al. presented a deep neural network-based face recognition system, *DeepFace* [55]. Furthermore, many other extended versions of *DeepFace* have been demonstrated in multiple studies [56–59]. Adjabi et al. thoroughly reviewed face recognition techniques and their comparisons and future scope in their study [51]. Kortli et al. surveyed popular face recognition techniques in all three categories, that is, local, holistic, and hybrid approaches, in their study [52]. They compared these techniques in terms of accuracy, complexity, and robustness. They also discussed the advantages and disadvantages of the respective approaches. Wang et al. efficiently surveyed DL-based face recognition techniques in their study [60]. They exhaustively reviewed various popular DL-based approaches, including autoencoder-based, CNN-based, and GAN-based techniques. They also enumerated the key features, advantages, and disadvantages of these techniques. Furthermore, they summarized some of the commonly used datasets for deep face recognition. Moreover, they indexed the emerging real-world issues and major technical key challenges in deep facial recognition.

However, an application involving person identification must address important privacy concerns [61]. In particular, facial identification in the public domain must tackle individual freedom and ethics-related issues [51, 62]. Therefore, the state-of-the-art research problem in face recognition is the reidentification of an individual on anonymized data. Rocher et al. demonstrated the likelihood of correctly reidentifying a specific individual, even with the anonymized dataset [30]. They suggested a generative graphical model that can be trained on incomplete data to accurately identify individuals. Rooijen et al. suggested 2D video tracking for the reidentification of individuals in an anonymized dataset [20]. They suggested that the real facial information of a person is not necessary for reidentification. Luo et al. suggested effective training tricks for person reidentification [63]. A residual learning framework using the residual network (*ResNet*) model was suggested in [64] for visual recognition tasks. This facilitated the easier and more efficient training of a substantially deeper network. Schroff et al. suggested unified embedding using only 128 bytes per face for efficient face recognition [65]. They developed their network by incorporating the batch input layer and deep CNN, followed by normalization. They used triplet loss to minimize the training errors. The world's simplest face recognition library (Dlib face recognition) is a popular and efficient tool for extracting facial landmarks [66]. It is a cross-platform open-source machine-learning toolkit that supports the development of machine-learning algorithms. It helps in recognizing and manipulating faces. Intent and behavior have been successfully detected using various techniques. Facial gesture sensing is performed using virtual reality (VR) and augmented reality (AR) devices, respectively in [67, 68]. AR/VR devices provide sensor responses to detect the intent or behavior of the user. However, FAV in-cabin monitoring requires intent or behavior detection using visual (computer vision (CV)-based) monitoring approaches.

2.2. Method. In this study, we proposed a representation learning-based approach to generate the identity signature of occupants. This signature is capable of deidentifying a person concerned with an irregular situation in the cabin of level 4 and beyond AVs. We proposed facial anonymization and reidentification system to provide countermeasures in case of an irregular situation. Therefore, this method provides personal information security with traces of the concerned person in case of any abnormality. The proposed method includes four main tasks. First, face anonymization with reenrollment. This is performed by using the face agnostic face swapping technique. It uses a set of GANs. These GANs are used for three purposes: facial reenactment and segmentation, facial inpainting, and facial blending. After accomplishing face anonymization, the second task is to extract the facial identity features of the occupant's faces in pairs (before and after anonymization, i.e., ID_R and ID_A) using the *ResNet*-based model. These *IDs* are stored in the cloud, and the anonymized video frames of in-cabin monitoring are transferred to the data center via the cloud for further processing. The third task of the proposed approach is to identify the accused by identity feature matching. Similarity matching of the *ID* of the accused obtained at the data center with the *IDs* of the occupants stored in the cloud ensures the identification of the concerned person (ID_A). However, it is the *ID* of the anonymized face of the accused. The Euclidean distance metric was used for similarity matching. Similarly, using the stored pairs of *IDs* (IDR and ID_A), we can obtain the real face identity feature of the accused (ID_R). Finally, in the fourth task, the evidence of the accused is obtained by matching the similarities between the *IDs* of the suspects with the *ID* of the accused during an investigation. Further details of the proposed method are provided in the following sections.

2.2.1. Facial Identity Feature Vector. The facial identity feature is (128, 1)-dimensional encoding of a facial image. It contains the encoded landmarks of the face using the *ResNet* model. The FaceNet-based CNN model and Facedlib face recognition library are used to extract the 128D identity features (*ID*) from the faces. Additionally, 128D is optimal embedding, which results in appropriate features required for reidentification or measuring the similarity between two faces. It has already been validated in the "*FaceNet*" architecture that fewer than 128D identity features deteriorate the identification performance; however, increasing the dimension only unnecessarily increases the number of parameters. This is the main reason for adopting the 128D identity features for recognizing faces.

Figure 3 shows the (128, 1)-dimensional facial identity feature vector generation of the occupant's face image. It uses a *ResNet*-based architecture consisting of 29 convolutional layers for this purpose. The *ResNet* architecture facilitates the dipper layer accessibility. Additionally, they have an inherent tendency to minimize the training error loss by increasing the number of layers. The triplet loss function is used to estimate the error in the reidentification of the concerned person. It performs similarity matching on the 128D identity features. For the anonymized anchor image *ID* (I_A), positive anonymized image *ID* (I_P), and negative anonymized image *ID* (I_N), the triplet loss is estimated by the following equation:

$$\mathscr{L}(A, P, N) = \max\left(\|I_A, I_{P2}\| - \|I_A, I_{N\|2} + \text{margin}, 0\right). \quad (1)$$

The anonymized anchor image ID (IA) represents the 128D ID of the person figured out in an irregular situation. The positive anonymized image (IA) is the stored image 128D ID of the same person on the cloud, and the negative anonymized image ID (IA) is the 128D ID of another occupant. Here, ($\|x, y\|_2$) denotes the "Euclidean distance" between pairs $\{x, y\}$ in the triplet loss function. A factor margin is included in equation (1) to reduce the chances of misclassification. These facial features are incorporated in 128D encoding and are used as the facial recognizer using only 128 bytes per face.

Furthermore, a distance-based classifier compares the 128D features to identify the person involved in an irregular situation. It represents the difference between two feature vectors in Euclidean space. Suppose that image (R) represents the person. Image (C) is the stored image (copy) of the same person on the cloud, and image (D) is an image of another occupant. Further, $f(x)$ represents the 128D encoding of the image $f(x)$. The similarity (S) in the vector space is measured by the following equation:

$$S = \min\left(\|f(R), f(C)\|_2, \left(\|f(R), f(D)\|_2\right)\right). \quad (2)$$

It guarantees that images (R) and (C) are of the same occupant and are different from image (D), which is the image of another occupant.

2.2.2. Source Image Generation. A source image was required for face swapping in facial anonymization. It is used to replace the face appearing in the target image. This replacement, that is, swapping, should produce a realistic result that seamlessly reenacts the anonymized face that is similar to the target face. Our recommendation is to use a nonreal face as the source image. It mitigates any chaos/conflicts that may occur by using any real face as the source image. Therefore, in our proposed method, we used GAN-

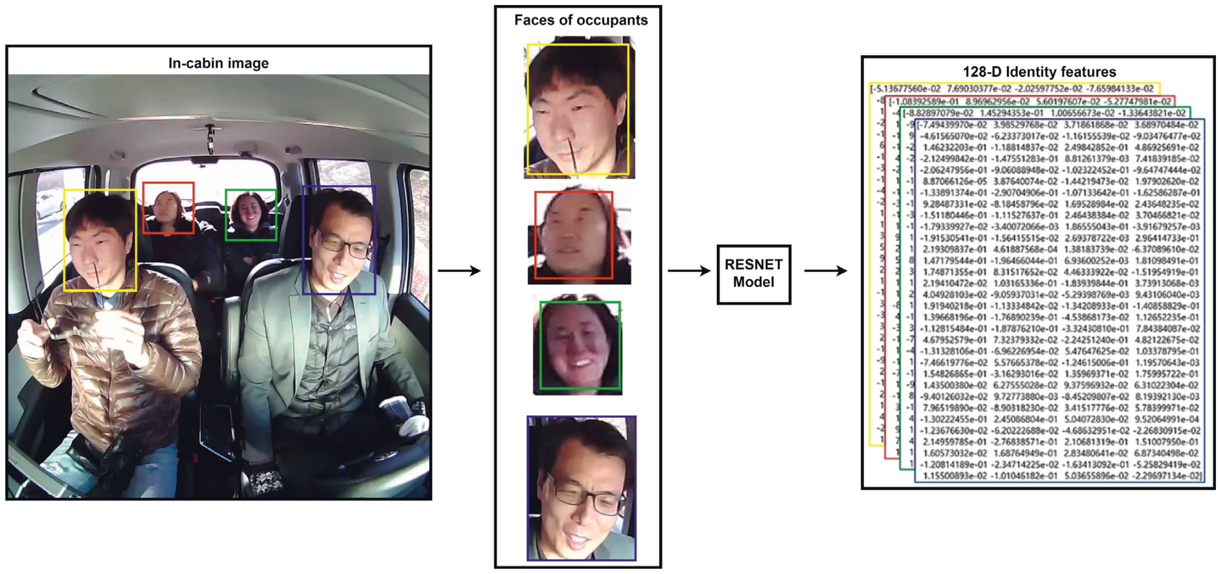


Figure 3: Illustration of 128D facial identity feature vector generation (from the occupant's face image). Image shown is taken from our in-cabin monitoring database. The numerical values in the yellow, red, green, and blue colored boxes are representing respective passengers' (128, 1)-dimensional facial identity feature vectors (*ID*).

generated virtual human faces as the source image. We have considered generating appropriate source faces that can effectively render the original emotions or behaviors performed by the occupants. It helps in further event and behavior-monitoring tasks. Figure 4 shows the proposed source image generation process. We applied the concept of similarity matching in vector space to select a similar source face for each target face from the set of virtual human faces (nonreal face as the source image). Similarity matching between source and target faces facilitates reciprocating similar emotions and intents, which is necessary for further monitoring applications.

Figure 4 shows the source image generation process. The face detector detects the faces (target faces) of the occupants (from the in-cabin visual input). The identity feature extractor extracts the *IDs* (128D identity features) of faces (target faces) and matches the similarity of the target faces with the set of virtual human faces (source faces) to find the most appropriate source face. This similarity matching is in the vector space (Euclidean distance matching between the extracted face *ID* and *IDs* of the set of virtual human faces).

2.2.3. Facial Anonymization.

Facial anonymization requires exactitude in the anonymized faces to mitigate errors in further processing. Therefore, swapping should be performed efficiently to provide unaltered expressions and emotions over the anonymized face. We used the concept of FSGAN for facial anonymization to provide personal privacy during in-cabin monitoring of irregular situations. This requires perfection in the following three tasks:

(i) Facial Reenactment and Segmentation. To obtain proper facial swapping, we must estimate the proper reenacted face. This is performed by the proper segmentation of the face and hair segments of the target image. Proper facial reenactment requires separate face and hair segmentations with the mapping of 2D facial landmark positions. Therefore, the stepwise loss function is considered as the objective function for implementing facial reenactment. For ith layer feature map ($F_i \in \mathbb{R}^{C_i \times H_i \times W_i}$), the perceptual loss ($\mathscr{L}_{\text{perc}}$) between pairs of images (x, y) is expressed as follows:

$$\mathscr{L}_{\text{perc}}(x, y) = \sum \frac{1}{C_i \times H_i \times W_i} \times \|F_i(x), F_i(y)\|_2. \tag{3}$$

The reconstruction loss ($\mathscr{L}_{\text{rec}}$) between a pair of images (x, y) is expressed as follows:

$$\mathscr{L}_{\text{rec}}(x, y) = \lambda_{\text{perc}} \times \mathscr{L}_{\text{perc}}(x, y) + \lambda_{\text{pixel}} \times \mathscr{L}_{\text{pixel}}(x, y), \tag{4}$$

where "λ" is the corresponding hyperparameter ($\lambda_{\text{perc}} = 1$; $\lambda_{\text{pixel}} = 0.1$; $\lambda_{\text{adv}} = 0.001$; $\lambda_{SG} = 0.1$; $\lambda_{\text{rec}} = 1$; $\lambda_{\text{stepwise}} = 1$) and $\lambda_{\text{reenactment}}$ is linearly increased from 0 to 1 during training. Pixelwise loss ($\mathscr{L}_{\text{pixel}}$) between a pair of images (x, y) is calculated as ($\mathscr{L}_{\text{pixel}}(x, y) = \|x - y\|$). We have used the multiscale discriminator adversarial loss objective function to improve the realism of the generated images. The adversarial loss ($\mathscr{L}_{\text{adv}}$) between the generator and discriminator (G, D) is expressed as follows:

$$\begin{aligned} &\mathscr{L}_{\text{adv}}(G, D) = \min\Big(\max\Big(\sum \mathscr{L}_{\text{GAN}}(G, D)\Big)\Big),\\ &\mathscr{L}_{\text{GAN}}(G, D) = E_{(x,y)}[\log D(x, y)] + E(x)[\log(1 - D(x, G(x)))], \end{aligned} \tag{5}$$

where "$E_{(x,y)}$" is the expected value over all real data instances. "$E_{(x)}$" is the expected value over all random inputs to the generator. The reenactment generator loss ($\mathscr{L}_{RG}$) is given by the following equation:

$$\mathscr{L}_{RG} = \mathscr{L}_{\text{perc}} + \mathscr{L}_{\text{rec}} + \mathscr{L}_{\text{adv}}. \tag{6}$$

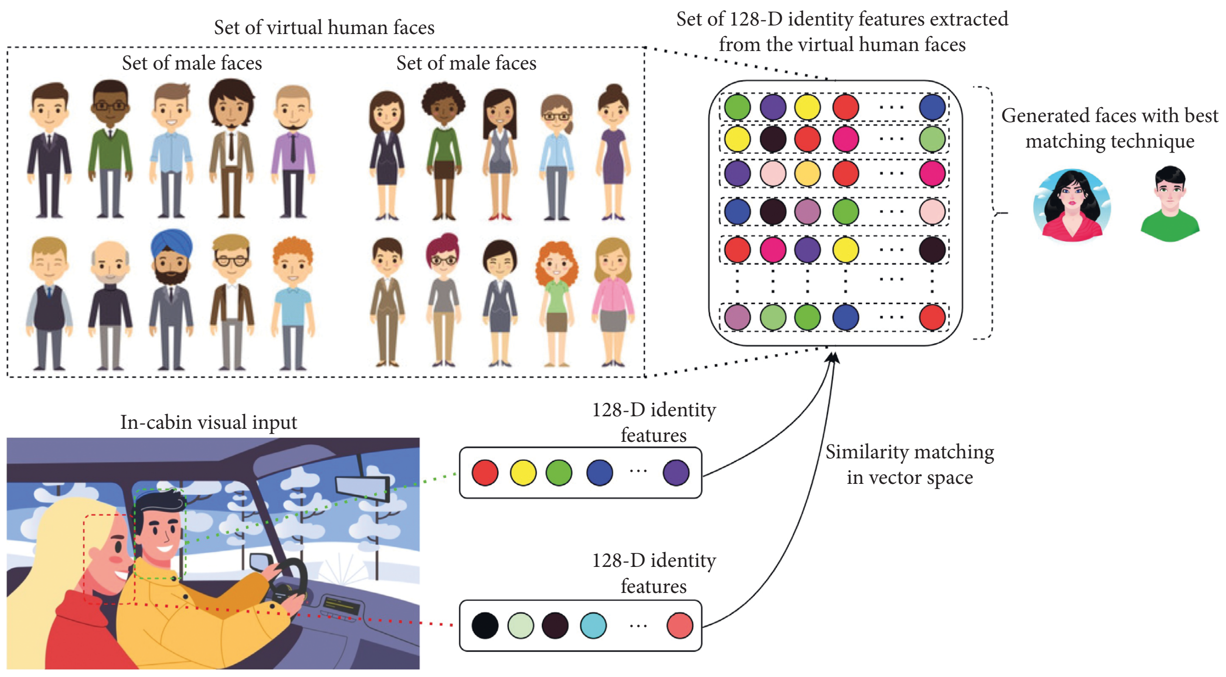

Set of virtual human faces
Set of male faces
Set of male faces
Set of 128-D identity features extracted from the virtual human faces
Generated faces with best matching technique
In-cabin visual input
128-D identity features
Similarity matching in vector space
128-D identity features

Figure 4: Source image generation using AI-generated faces with the best matching technique.

The perpetual loss is used to estimate the errors in capturing fine facial details, and the reconstruction loss is used to evaluate pixelwise color inaccuracy. Adversarial loss improves the generated images and provides a realistic look. The standard cross-entropy loss ($\mathscr{L}_{CE}$) is defined as (for truth label "t_i" and the "*SoftMax*" probability "P_i" for i^{th} class)

$$\mathscr{L}_{CE} = -\sum t_i \times \log(P_i). \tag{7}$$

Further, segmentation generator loss ($\mathscr{L}_{SG}$) is obtained by the following equation:

$$\mathscr{L}_{SG} = \mathscr{L}_{CE} + \mathscr{L}_{\text{pixel}}. \tag{8}$$

(ii) Facial Inpainting. This method estimates the missing portions of the reenacted face based on the face and hair segmentation of the target image. The inpainting generator loss ($\mathscr{L}_{IP}$) was calculated using the following equation:

$$\mathscr{L}_{IP} = \mathscr{L}_{\text{rec}} + \mathscr{L}_{\text{adv}}. \tag{9}$$

(iii) *Facial Blending.* It blends the completely reenacted face such that the swapped face matches the background environment like the original target face. The loss function ($\mathscr{L}_B$) for facial blending is obtained using the following equation:

$$\mathscr{L}_B = \mathscr{L}_{\text{perc}} + \mathscr{L}_{\text{adv}}. \tag{10}$$

The identity signature is generated corresponding to each occupant (a pair of identity signatures for real and anonymized faces) in the FAV. After facial anonymization, the video frames are transmitted to the cloud along with a pair of identity signatures of the occupants.

2.2.4. Anonymized Person Reidentification in Abnormal Situations. The proposed IMS facilitates the reidentification of the person involved in an abnormal situation. In our algorithm, in-cabin facial anonymization for preserving identity before transmitting the video frames to the cloud was achieved through the following pseudocode. The identity signature is generated corresponding to each occupant in the FAV. It is a vector of size 1×128. Therefore, for each occupant, we have a pair of identity signatures corresponding to the original and anonymized faces. Each pair is stored in the cloud. In any irregular situation, the concerned person is back-traced by matching the identity signature and anonymized face. The following is Pseudocode 1 of our proposed approach for obtaining the identity features (*ID*) of the person involved in an abnormal situation.

We considered virtual human face generation for the source faces. These faces are used to swap the target face in the captured visual in-cabin dataset. The source faces are generated depending on the similarity of the target face in the vector space. A similar source face provides the exactitude in replaying the facial gestures. This facilitates better reenactment performances. The concept of virtual human face generation for the source face protects any chaos or risk of threatening others' identities. Furthermore, we generated the facial identity signatures of the original and anonymized faces. These identity signatures help backtrack the concerned person in the event of an irregular situation. The identity signature is only vectored information. In other words, the identity signature in our proposed approach is extracted from a face that is used to reidentify the face. However, a face cannot be recreated using this information. Therefore, personal identity is not revealed through the identity signature. Our proposed approach provides proof or evidence that confirms the identity of the concerned person. The

```
(i) Definitions: Faces of the occupant ($F$); target face ($T$); appropriate source face ($S$); anonymized face ($A$); identity features ($ID$):
    real face ID ($ID_R$); anonymized face ID ($ID_A$); ID of the occupant that caused an abnormal situation ($ID_{A\_AS}$); and an in-cabin
    abnormal situation ($AS$).
(ii) Functions: $\mathbb{F}$ = face detector; $\mathbb{S}$ = source detector; $\mathbb{A}$ = anonymizer; $\mathbb{I}$ = $ID$extractor.
(iii) Input: video frames (in-cabin)
(1)      for $i = 1$ to range of the occupant:
(iv)             $T(i) = \mathbb{F}$ (Input)
(v)              $S(i) = \mathbb{S}(T(i))$ # search most similar source face for target face
(vi)             $A(i) \leftarrow \mathbb{A}(T(i), S(i))$
(vii)            $ID_R(i) = \text{II}(T(i))$ # 128D feature vector of the target face $ID_A(i) = \text{II}(A(i))$
(2)      store: $ID(i) \leftarrow (ID_R(i); ID_A(i))$
(3)      At datacenter: monitor event and behavior for $AS$:
(viii)           if occupant $j$ is involved in $AS$, then:
(xi)                     generate $(ID_{A\_AS}(j))$ #$ID$ of $j^{th}$ occupant in abnormal situation
(x)                      match $ID$:
(xi)                             for $ID$ from 1 to range of the $ID$:
(xii)                                    $k = argmin(\|ID_{A\_AS}(j), ID_A(:)\|_2)$
(4)      Map: $ID_R(k) \leftarrow ID_A(k)$
(xiii) return $(ID_R(k))$ # the algorithm returns the real face ID of an anonymized person
```

Pseudocode 1: Algorithm for obtaining the ID of a person involved in an abnormal situation.

```
(i) Definitions: Target face (occupant's face) captured during the investigation ($T_{inv}$); $ID$ of the occupant's face obtained during an
    investigation ($ID_{inv}$); $ID$ of the person involved in the abnormal situation ($ID_R$); and real face of the occupant involved in the
    abnormal situation ($O$).
(ii) Functions: II = ID extractor.
(iii) Input: $T_{inv}$; $ID_R$
(1)          At investigation:
(iv)                 for $i$ from 1 to range of the target faces:
(v)                  $ID_{inv}(i) = \text{II}(T_{inv}(i))$
(vi)                 match $ID$: # compare $ID_R$ and the suspect face $ID$
(vii)                        $j = \text{argmin}_i(\|\text{ID}_R, \text{ID}_{inv}(i)\|_2)$
(2)          Map: $O \leftarrow j$
(viii) return $(O)$
```

Pseudocode 2: Algorithm for evidence of the person involved in the abnormal situation.

following is Pseudocode 2 of our proposed approach for evidence of the person involved in an abnormal situation.

In the case of proof or evidence, our method determines who is the concerned person. The returned identity feature (real face $ID_R(k)$) in Pseudocode 1 refers to the crucial identity parameter of the person involved in an abnormal situation. Matching the identity feature at the time of investigation with the obtained ID (real face $ID_R(k)$) confirms the person involved in an abnormal situation. Therefore, this approach easily locates the person involved in an irregular situation without any breach of others' identities.

3. Results and Discussion

In our experiment, we first anonymized the occupants of the FAV to secure their privacy in the public domain. Further, we applied the concept of vector space similarity to match the representation learning-based identity features for face recognition to locate the person involved in an irregular situation. The augmentation of the representation-learning-based identity feature introduces a new domain in reidentification. The proposed system was introduced to maintain personal privacy during the monitoring. We examined our proposed system for the in-cabin monitoring task of the FAV. We captured our database for in-cabin monitoring in abnormal situations. The similarity measure ($S_{i,j}$) is calculated by the Euclidean distance (ED) metric that is expressed as follows:

$$S_{i,j} = \|f(i), f(j)\|_2, \tag{11}$$

where $f(i)$ and $f(j)$ represent the 128D encoding of images i and j, respectively. Therefore, the similarity measure identifies the distance (Euclidean distance) between two pairs of IDs (128D encoding). The lesser the distance is, the closer the faces are.

3.1. Appropriate Source Faces.

We proposed the concept of an appropriate source face in our facial anonymization approach. For every occupant face (target face), an appropriate source face is obtained by matching their similarity in the vector space. We considered various scenarios to assess the efficacy of our proposed approach, including single and multiple faces in the input image frame. Figure 5 shows the complete set of the considered source faces in our experiment. We considered a set of 24 source faces (shown below). All these faces were not real (AI-generated). The source faces were used to swap the target face in the facial anonymization process.

These faces are nonreal virtual human faces. Generated Photos provides GAN-generated faces, which are human faces of nonreal humans. This has the benefit of further augmentation in anonymization. We considered various scenarios in our experiments. Examples include images with a single face only (for both males and females), multiple faces for males only and females only, and multiple faces for both males and females. These are in-cabin images obtained from the public domain (through an image search on the web) and are shown in Figure 6. We considered different scenarios for the occupants in the cabin. Therefore, in F1 and F2, there is only a single person in-cabin (F1: male and F2: female). In other scenarios, we considered more than one person in the cabin (only males, only females, and both males and females). Finally, we considered a family with children. There are four most appropriate source faces (S1 to S4) chosen for face anonymization.

Table 1 presents the similarities (in vector space) between the source and target faces, as shown in Figures 5 and 6.

These values follow the facial similarities of the source and target faces. These values measure the distance between the identity features of the source and target faces. The lower the values are, the more similar the faces are. The values in the green boxes represent the minimum Euclidean distances. These minimum values indicate appropriate source faces for anonymization. We can observe that the male target faces have lesser distances for male source faces than for female source faces. Interestingly, the distance values follow the similarity in looks as well. The eastern looks target faces have a lesser distance for eastern source faces than for the western source face, and vice versa. Female source faces have a lesser distance than the identity features of children's target faces.

3.2. Privacy Preservation during In-Cabin Monitoring. Facial anonymization is performed after deciding the appropriate source face using FSGAN-based face swapping and reenactment. Figure 7 depicts the reenacted anonymization of the target faces. Here, the first row (F1 to F8) and the third row (F9 to F23) show the original in-cabin visual inputs, and the corresponding anonymized output is represented in the second row (A1–A8) and fourth row (A9–A23). We chose four source faces (S1 to S4 shown in Figure 6) to swap the target faces (F1 to F23).

It is evident from this result that perfect reenactments are achieved even in the anonymized domain. Thus, it discerns the preservation of personal privacy during monitoring and surveillance operations. Furthermore, this appropriate reenactment supports the detection of abnormal or irregular situations in real time. To examine abnormality detection in the anonymized domain, we have experimented by considering vandalism as an irregular situation inside the vehicle cabin. We created our database for a similar situation. Snippets of the vandalism inside the vehicle are shown in Figure 8. We created a situation wherein occupants in the back seat of the vehicle started fighting with the occupants in the front seat. Four scenes were captured in our experiment. Shoulder shaking is shown in scene #1. Scene #2 shows a slapping scenario. Head shaking is discerned in scene #3, and scene #4 represents a neck choking incident inside the cabin of the vehicle. The identity features (*ID*s) of each occupant were calculated for normal and irregular situations. It is clearly observed that O3 (in the green box) is responsible for the irregular situation (in-cabin vandalism of the vehicle shown in the red box).

3.3. Person Reidentification in Abnormal Situations. Table 2 presents the similarities of the anonymized identity feature (ID_A) with the anonymized facial identity feature of occupant #3 (ID_{A_IS}). Here, ID_{A_IS} is the anonymized identity feature of the occupant who is involved in an irregular situation calculated at the data center, and ID_A is the anonymized identity feature of the occupant stored in the cloud.

The values in the green boxes represent the minimum Euclidean distances. These minimum differences between the *ID*s indicate the involved person. The original *ID* of this person is stored in the cloud. Therefore, by mapping the *ID*, we can easily identify the real person. Reidentification was performed by backtracking the *ID* obtained from the cloud and pictures of the occupants taken during the investigation. The *ID* of the person involved in an abnormal situation from the cloud (ID_R) needs to be matched with the *ID*s of the occupants inside the vehicle for facial identification of the person. This approach provides proof or evidence confirming the identity of the concerned person. For assurance of the person involved in the abnormal situation, we took pictures of the occupants (during an investigation). The images are shown in Figure 9. Now, the identity feature of each occupant is extracted to match the concerned person *ID* (ID_R) (as per Pseudocode 2). First, we compared the similarity between the faces of the occupants inside the vehicle with those of the other faces captured during the investigation. This is required to ensure that the occupants are the same.

Table 3 presents the similarity measures between the occupants' *ID*s extracted during an investigation and their *ID*s extracted from the in-cabin images.

The minimum Euclidean distances are represented by the green boxes. Here, minima indicate that the occupants O and O'' are the same. Thereafter, assurance of the involved person is performed by matching the identity feature of the occupants extracted from the in-cabin image of the vehicle with the *ID* of the person involved in an abnormal situation (stored in cloud ID_R). Table 4 presents the similarity measures between the occupants' *ID*s extracted from the in-cabin image with the obtained *ID* of the person involved in an abnormal situation (stored in cloud ID_R).

FIGURE 5: Set of virtual human faces (AI-generated faces). These virtual human faces are obtained from Generated Photos. It provides AI-generated images that are free from any copyrights, distribution rights, and infringement claims (source: Generated Photos (https://generated.photos/)).

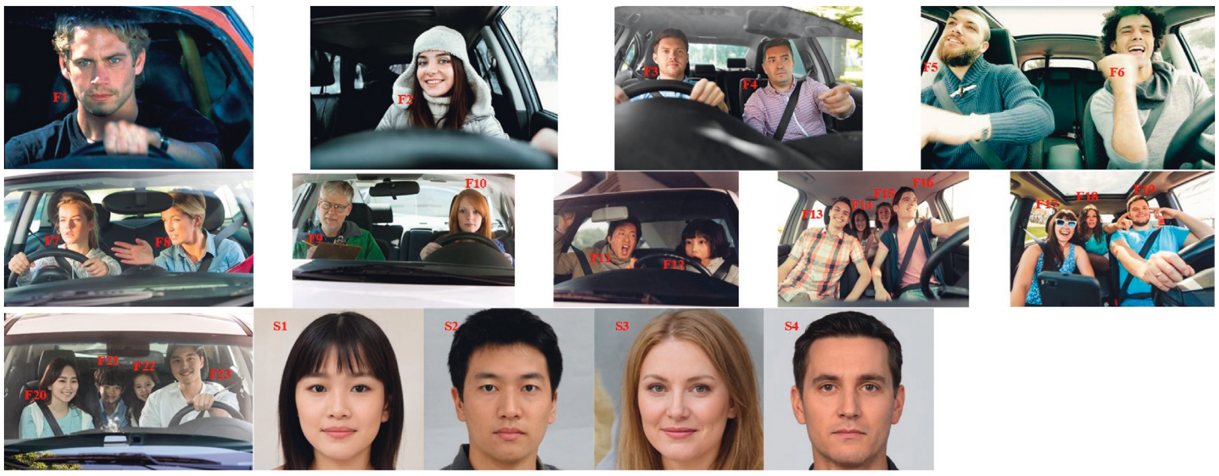


FIGURE 6: We have chosen single and multiple faces in the input images in different scenarios: single face (only male or only female), multiple faces (only male), multiple faces (only female), and multiple faces (both male and female). Here, the target (occupant) faces are indexed from F1 to F23, and considered source faces (both male and female) are indexed from S1 to S4.

TABLE 1: Similarities between the source and target faces.

Scenario		Target (occupants[†])	Similarity measure (using Euclidean distance)			
			S1	S2	S3	S4
Single face	Male	F1	0.91489481	0.80287961	0.89433056	**0.74069120**
	Female	F2	0.78818484	0.81636149	**0.68592050**	0.76422129
Multiple face	Male	F3	0.91414391	0.88400388	0.87615788	**0.83486502**
		F4	0.79685733	**0.71862379**	0.93450242	0.75311709
	Male	F5	0.91205174	0.82094296	0.87266242	**0.78036428**
		F6	0.81236709	0.80381698	0.93859941	**0.67143296**
	Female	F7	0.81097788	0.82709409	**0.71891988**	0.86480495
		F8	0.85947196	0.78512872	**0.77978978**	0.90500906
	Both	F9	0.89428390	0.83158545	0.88401185	**0.80949051**
	Both	F10	0.84977716	0.90480697	**0.71174311**	0.94153902
	Both	F11	0.65831500	**0.52838455**	0.95610671	0.88142326
	Both	F12	**0.38916649**	0.45361382	0.89109294	0.88496564
	Both	F13	0.79624321	0.80097813	0.88202307	**0.71975099**
	Both	F14	**0.63660264**	0.67343593	0.88004248	0.96042187
	Both	F15	0.84524707	0.89008615	**0.77500429**	0.86828727
	Both	F16	0.74547080	0.77676084	0.93155677	**0.73583944**
	Both	F17	0.79179192	0.80390987	**0.73040828**	0.88839209
	Both	F18	0.78950908	0.79986798	**0.72049968**	0.94658813
	Both	F19	0.90007099	0.85322199	0.99829307	**0.85322199**
	Both	F20	**0.40197132**	0.63806865	0.83032199	0.88047087
	Both	F21	**0.46230089**	0.53098278	0.85879277	0.86241199
	Both	F22	**0.48055832**	0.54356384	0.83216505	0.81304802
	Both	F23	0.55751665	**0.45767183**	0.90473598	0.78517239

[†]The occupants are numbered from left to right clockwise.

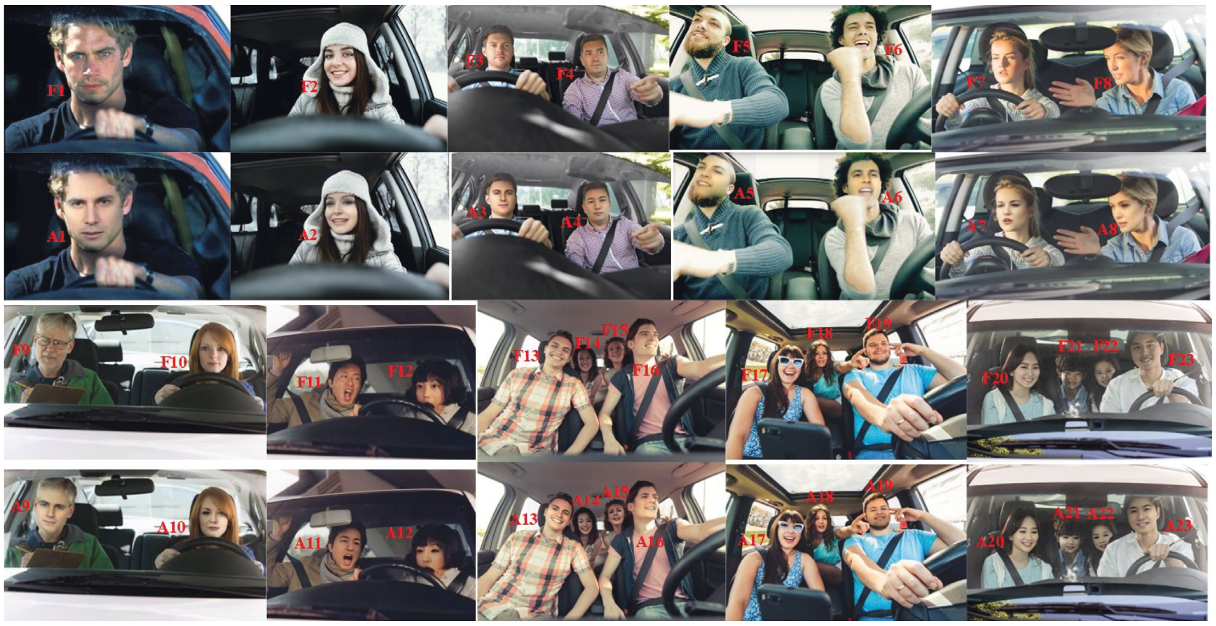


Figure 7: Facial anonymization with reenactment. F1 to F23: original images. A1 to A23: corresponding anonymized images considering appropriate source faces.

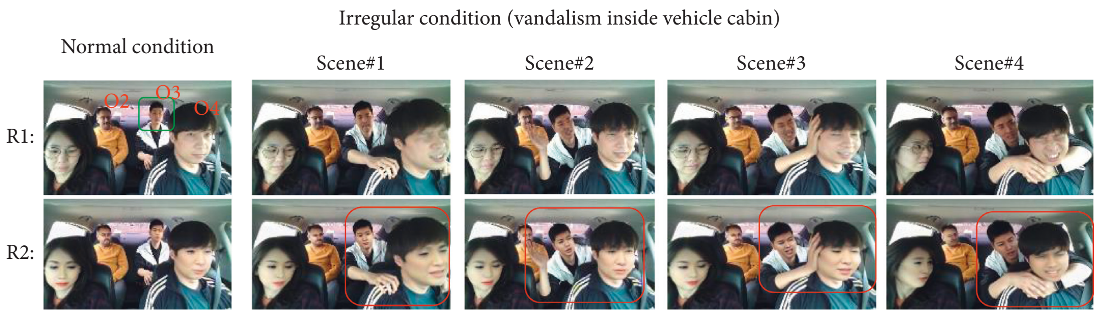


Figure 8: Snippets of our database showing vandalism inside the vehicle cabin. The original image under normal and irregular situations is in row R1, and the corresponding anonymized images are shown in row R2. The occupants are numbered from left to right clockwise (O1, O2, O3, and O4). Scene #1: O3 shakes shoulder of O4; scene #2: O3 tries to slap O4; scene #3 O3 shakes head of O4; and scene #4: O3 chokes neck of O4. Green box: concerned person and red box: in-cabin vandalism.

Table 2: Identity feature matching between ID_{A_IS} #3 at the data center and other stored *ID*s of the occupants in the cloud for different scenarios.

Scene	Similarity measure (in Euclidean distance)			
	ID_A #1	ID_A #2	ID_A #3	ID_A #4
Scene #1	0.52893346	0.78363186	**0.35424358**	0.42124692
Scene #2	0.45234707	0.79687774	**0.35880417**	0.40963131
Scene #3	0.49863882	0.77615540	**0.41716736**	0.44500655
Scene #4	0.74701755	0.5816643	**0.53716927**	0.73605501

Detail description of scenes (scenes #1–#4) is mentioned in Section 3.2.

The zero value in the green box indicates that the occupant (O3) is the person involved in an abnormal situation. Overall, this approach focuses on in-cabin monitoring with personal privacy preservation to avoid abnormal situations. Personal privacy preservation is achieved by using the concept of event and behavior monitoring in an anonymized domain. The person's reidentification is only for providing evidence in cases where the involved person is denying it.

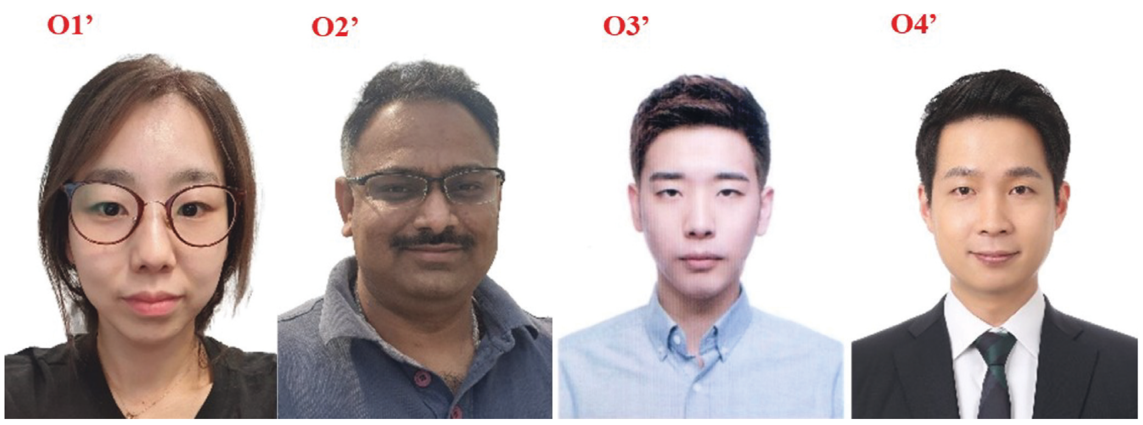


Figure 9: Other pictures of the occupants during investigation for matching. The numbering is the same as those in the in-cabin images from left to right (O1'', O2'', O3'', and O4'').

Table 3: Identity feature matching between the occupants' *ID*s extracted during the investigation and their *ID*s extracted from in-cabin images.

Occupant's *ID* (in-cabin)	Occupant's *ID*s were extracted during an investigation			
	ID_{O1}'	ID_{O2}'	ID_{O3}'	ID_{O4}'
ID_{O1}	**0.48432609**	0.79061829	0.72076523	0.69776956
ID_{O2}	0.75859866	**0.66392154**	0.79348079	0.72322158
ID_{O3}	0.52265982	0.77696899	**0.36226176**	0.47469558
ID_{O4}	0.64218059	0.81529880	0.52173335	**0.42871777**

Table 4: Identity feature matching between ID_R stored in the cloud with other occupant's *ID*s extracted from the in-cabin of the vehicle.

ID (person involved)	*IDs of the occupants (In-cabin)*			
	ID_{O1}	ID_{O2}	ID_{O3}	ID_{O4}
ID_R	0.62511349	0.75967812	**0**	0.52087737

4. Conclusions

Identity feature augmentation in anonymization is a potential solution for providing privacy in public domain monitoring. Identification of the involved person is crucial, especially in abnormal situations. The proposed intelligent IMS augments the security features with privacy. This method is suitable for creating a monitoring database without any restrictions or legalities. We performed various scenarios to assess the efficacy of the proposed system. It provided an efficient algorithm to perform monitoring tasks in the public domain without any threat to the personal identity of a person. This helped in reidentification, even with an anonymized face. In the future, this algorithm can be implemented on various public domain monitoring platforms, such as transportation systems, shopping centers, theaters, hospitals, highways, fuel refilling stations, smart city applications, and toll plazas.

Data Availability

The image data used to support the findings of this study are included in this paper.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Authors' Contributions

Ashutosh Mishra and Shiho Kim contributed to the conceptualization. Ashutosh Mishra, Jaekwang Cha, and Shiho Kim developed the methodology, performed formal analysis, and investigated. Ashutosh Mishra reviewed and edited the paper. Ashutosh Mishra and Jaekwang Cha provided the software and performed validation, visualization, and data curation. Ashutosh Mishra provided the resources and preparation. Shiho Kim contributed to supervision, project administration, and funding acquisition. All authors have read and agreed to the published version of the paper.

Acknowledgments

This study was partially supported by the Brain Pool Program through the National Research Foundation of Korea

(NRF) funded by the Ministry of Science and ICT (NRF-2019H1D3A1A01071115) and by the Institute for Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korean government (MSIT) (no. 2021-0-01352, Development of Technology for Validating the Autonomous Driving Services from the Perspective of Laws and Regulations). The authors thank all the volunteers for their valuable contribution to our database creation.

References

[1] P. Vennam, T. C. Pramod, B. M. Thippeswamy, Y.-G. Kim, and B. N. Pavan Kumar, “Attacks and preventive measures on video surveillance systems: a review,” *Applied Sciences*, vol. 11, no. 12, Article ID 5571, 2021.

[2] J. Janai, F. Güney, A. Behl, and A. Geiger, “Computer vision for autonomous vehicles: problems, datasets and state of the art,” *Foundations and Trends in Computer Graphics and Vision*, vol. 12, no. 1–3, pp. 1–308, 2020.

[3] “SAE international releases updated visual chart for its “levels of driving automation” standard for self-driving vehicles,” February 2021, https://www.sae.org/news/press-room/2018/12/sae-international-releases-updated-visual-chart-for-its-%E2%80%9Clevels-of-driving-automation%E2%80%9D-standard-for-self-driving-vehicles.

[4] “Automated vehicles for safety,” 25 February 2021, https://www.nhtsa.gov/technology-innovation/automated-vehicles-safety.

[5] A. Mishra, J. Kim, D. Kim, J. Cha, and S. Kim, “An intelligent in-cabin monitoring system in fully autonomous vehicles,” in *Proceedings of the 2020 International SoC Design Conference (ISOCC)*, pp. 61-62, Yeosu, Korea, October 2020.

[6] “UK’s facial recognition technology ‘breaches privacy rights,” February 2021, https://www.theguardian.com/technology/2020/jun/23/uks-facial-recognition-technology-breaches-privacy-rights.

[7] “Facial recognition technology privacy and accuracy issues related to commercial uses,” February 2021, https://www.gao.gov/assets/710/708045.pdf.

[8] “Facial recognition technology fundamental rights considerations in the context of law enforcement,” February 2021, https://fra.europa.eu/en/publication/2019/facial-recognition-technology-fundamental-rights-considerations-context-law.

[9] P. Climent-Pérez and F. Florez-Revuelta, “Protection of visual privacy in videos acquired with RGB cameras for active and assisted living applications,” *Multimedia Tools and Applications*, vol. 80, no. 2, pp. 23649–23664, 2021.

[10] F. Bignami, *Schrems II: The Right to Privacy and the New Illiberalism*, Verfassungsblog, Germany, 2020.

[11] D. Dushi, *The Use of Facial Recognition Technology in EU Law Enforcement: Fundamental Rights Implications*, Global Campus Open Knowledge Repository, Oxford, UK, 2020.

[12] A. Mekrani, *The Future of Facial Recognition in Relation to Privacy,” Master Thesis*, Tilburg University, Tilburg, Netherlands, 2020.

[13] D. Naranjo, *Your Face Rings a bell: How Facial Recognition Poses a Threat for Human Rights*, Global Campus Open Knowledge Repository, Oxford, UK, 2020.

[14] “How facial recognition technology threatens basic privacy rights,” February 2021, https://www.computerweekly.com/feature/How-facial-recognition-technology-threatens-basic-privacy-rights.

[15] M. Doktor, “Facial recognition and the fourth amendment in the wake of carpenter v. United States,” *University of Cincinnati Law Review*, vol. 89, no. 2, p. 552, 2021.

[16] A. Daly, “Privacy in automation: an appraisal of the emerging Australian approach,” *Computer Law & Security Review*, vol. 33, no. 6, pp. 836–846, 2017.

[17] M. Smith and S. Miller, “The ethical application of biometric facial recognition technology,” *AI & Society*, vol. 37, no. 1, pp. 1–9, 2021.

[18] R. Van Noorden, “The ethical questions that haunt facial-recognition research,” *Nature*, vol. 587, no. 7834, pp. 354–358, 2020.

[19] “Facial-recognition research needs an ethical reckoning,” July 2021, https://www.nature.com/articles/d41586-020-03256-7.

[20] A. V. Rooijen, H. Bouma, R. Pruim, J. Baan, W. Uijens, and J. V. Mil, “Anonymized person re-identification in surveillance cameras,” in *Proceedings of the Counterterrorism, Crime Fighting, Forensics, and Surveillance Technologies IV*, p. 115420A, International Society for Optics and Photonics, Edinburgh, UK, September 2020, http://toc.proceedings.com/56397webtoc.pdf.

[21] Y. Rong, C. Han, C. Hellert, A. Loyal, and E. Kasneci, “Artificial intelligence methods in in-cabin use cases: a survey,” 2021, http://arxiv.org/abs/2101.02082.

[22] F. S. Marcondes, D. Durães, F. Gonçalves, J. Fonseca, J. Machado, and P. Novais, “In-vehicle violence detection in carpooling: a brief survey towards a general surveillance system,” *Advances in Intelligent Systems and Computing*, vol. 1237, pp. 211–220, 2021.

[23] J. L. Bell, M. A. Taylor, G.-X. Chen, R. D. Kirk, and E. R. Leatherman, “Evaluation of an in-vehicle monitoring system (IVMS) to reduce risky driving behaviors in commercial drivers: comparison of in-cab warning lights and supervisory coaching with videos of driving behavior,” *Journal of Safety Research*, vol. 60, pp. 125–136, 2017.

[24] H. Szawarski, J. Le, and M. K. Rao, “Monitoring a vehicle cabin,” USPTO, Dallas, TX, USA, US. Patent 10252688, April 2019.

[25] X. Song, “Safety and clean vehicle monitoring system,” USPTO, Dallas, TX, USA, US. Patent 10196070, February 2019.

[26] A. Taeihagh and H. S. M. Lim, “Governing autonomous vehicles: emerging responses for safety, liability, privacy, cybersecurity, and industry risks,” *Transport Reviews*, vol. 39, no. 1, pp. 103–128, 2019.

[27] D. J. Glancy, “Privacy in autonomous vehicles,” *Santa Clara University School of Law*, vol. 52, no. 4, p. 1171, 2012.

[28] L. Collingwood, “Privacy implications and liability issues of autonomous vehicles,” *Information and Communications Technology Law*, vol. 26, no. 1, pp. 32–45, 2017.

[29] H. S. M. Lim and A. Taeihagh, “Autonomous vehicles for smart and sustainable cities: an in-depth exploration of privacy and cybersecurity implications,” *Energies*, vol. 11, no. 5, p. 1062, 2017.

[30] L. Rocher, J. M. Hendrickx, and Y. A. de Montjoye, “Estimating the success of re-identifications in incomplete datasets using generative models,” *Nature Communications*, vol. 10, no. 1, pp. 3069–9, 2019.

[31] A. Mishra, J. Cha, and S. Kim, “HCI based in-cabin monitoring system for irregular situations with occupants facial anonymization,” in *Proceedings of the International*

Conference on Intelligent Human Computer Interaction, pp. 380–390, Springer, Daegu, Korea, October 2020.

[32] T. Nakamura, Y. Sakuma, and H. Nishi, "Face-image anonymization as an application of multidimensional data k-anonymizer," *International Journal of Networking and Computing*, vol. 11, no. 1, pp. 102–119, 2021.

[33] S. Moschoglou, S. Ploumpis, M. A. Nicolaou, A. Papaioannou, and S. Zafeiriou, "3DFaceGAN: adversarial nets for 3D face representation, generation, and translation," *International Journal of Computer Vision*, vol. 128, no. 10-11, pp. 2534–2551, 2020.

[34] H. Hukkelås, R. Mester, and F. Lindseth, "DeepPrivacy: a generative adversarial network for face anonymization," in *Proceedings of the International Symposium on Visual Computing*, pp. 565–578, Springer, Lake Tahoe, NV, USA, September 2019.

[35] J. Dietlmeier, J. Antony, K. McGuinness, and N. E. O Connor, "How important are faces for person re-identification?," 2020, http://arxiv.org/abs/2010.06307.

[36] Q. Sun, A. Tewari, W. Xu, M. Fritz, C. Theobalt, and B. Schiele, "A hybrid model for identity obfuscation by face replacement," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 553–569, Munich, Germany, July 2018.

[37] V. Blanz, K. Scherbaum, T. Vetter, and H.-P. Seidel, "Exchanging faces in images," *Computer Graphics Forum*, vol. 23, no. 3, pp. 669–676, 2004.

[38] D. Bitouk, N. Kumar, S. Dhillon, P. Belhumeur, and S. K. Nayar, "Face swapping," *ACM Transactions on Graphics*, vol. 27, no. 3, pp. 1–8, 2008.

[39] Y. Zhang, L. Zheng, and V. L. Thing, "Automated face swapping and its detection," in *Proceedings of the IEEE 2nd International Conference on Signal and Image Processing (ICSIP)*, pp. 15–19, Singapore, December 2017.

[40] Y. Nirkin, I. Masi, A. T. Tuan, T. Hassner, and G. Medioni, "On face segmentation, face swapping, and face perception," in *Proceedings of the 13th IEEE International Conference on Automatic Face & Gesture Recognition*, pp. 98–105, Xi'an, China, 2018.

[41] T. Kim and J. Yang, "Selective feature anonymization for privacy-preserving image data publishing," *Electronics*, vol. 9, no. 5, p. 874, 2020.

[42] R. Natsume, T. Yatagawa, and S. Morishima, "RSGAN: face swapping and editing using face and hair representation in latent spaces," 2018, http://arxiv.org/abs/1804.03447.

[43] P. Korshunov, S. Marcel, and D. Fakes, "A new threat to face recognition? Assessment and detection," 2018, http://arxiv.org/abs/1812.08685.

[44] R. Natsume, T. Yatagawa, and S. Morishima, "FSNet: an identity-aware generative model for image-based face swapping," *Computer Vision - ACCV 2018*, vol. 11366, pp. 117–132, 2019.

[45] W. Bailer, "Face swapping for solving collateral privacy issues in multimedia analytics," in *Proceedings of the International Conference on Multimedia Modeling*, pp. 169–177, Thessaloniki, Greece, January 2019.

[46] Y. Nirkin, Y. Keller, and T. Hassner, "FSGAN: subject agnostic face swapping and re-enactment," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 7184–7193, IEEE, Seoul, Republic of Korea, August 2019.

[47] J. Naruniec, L. Helminger, C. Schroers, and R. M. Weber, "High -resolution neural face swapping for visual effects," *Computer Graphics Forum*, vol. 39, no. 4, pp. 173–184, 2020.

[48] A. K. Jain and S. Z. Li, *Handbook of Face Recognition*, Springer, New York, NY, USA, 2011.

[49] T. Huang, Z. Xiong, and Z. Zhang, *Face Recognition Applications, Handbook of Face Recognition*, pp. 371–390, Springer, New York, NY, USA, 2005.

[50] D. N. Parmar and B. B. Mehta, "Face recognition methods & applications," 2014, http://arxiv.org/abs/1403.0485.

[51] I. Adjabi, A. Ouahabi, A. Benzaoui, and A. Taleb-Ahmed, "Past, present, and future of face recognition: a review," *Electronics*, vol. 9, no. 8, p. 1188, 2020.

[52] Y. Kortli, M. Jridi, A. Al Falou, and M. Atri, "Face recognition systems: a Survey," *Sensors*, vol. 20, no. 2, p. 342, 2020.

[53] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.

[54] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, "Eigenfaces vs. Fisherfaces: recognition using class specific linear projection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 711–720, 1997.

[55] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "DeepFace: closing the gap to human-level performance in face verification," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1701–1708, IEEE, Columbus, OH, USA, September 2014.

[56] Y. Sun, X. Wang, and X. Tang, "Deep learning face representation from predicting 10,000 classes," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1891–1898, IEEE, Columbus, OH, USA, September 2014.

[57] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification," in *Proceedings of the 27th International Conference on Neural Information Processing Systems (MIT)*, pp. 1988–1996, Montreal, Canada, June 2014.

[58] Y. Sun, X. Wang, and X. Tang, "Deeply learned face representations are sparse, selective, and robust," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2892–2900, Boston, MA, USA, June 2015.

[59] Y. Sun, D. Liang, X. Wang, and X. Tang, "DeepID3: face recognition with very deep neural networks," 2015, http://arxiv.org/abs/1502.00873v1.

[60] M. Wang and W. Deng, "Deep face recognition: a survey," *Neurocomputing*, vol. 429, pp. 215–244, 2021.

[61] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft, "Privacy-preserving face recognition," in *Proceedings of the International Symposium on Privacy Enhancing Technologies Symposium,(LNSC)*, pp. 235–253, Seattle, WA, USA, August 2009.

[62] A. Roussi, "Resisting the rise of facial recognition," *Nature*, vol. 587, no. 7834, pp. 350–353, 2020.

[63] H. Luo, Y. Gu, X. Liao, S. Lai, and W. Jiang, "Bag of tricks and a strong baseline for deep person re-identification," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Long Beach, CA, USA, March 2019.

[64] 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 (CVPR)*, pp. 770–778, Las Vegas, NV, USA, June 2016.

[65] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: a unified embedding for face recognition and clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern*

Recognition (CVPR), pp. 815–823, Boston, MA, USA, October 2015.

[66] D. E. King, “Dlib-ml: a machine learning toolkit,” *Journal of Machine Learning Research*, vol. 10, pp. 1755–1758, 2009.

[67] J. Kim, J. Cha, and S. Kim, “Hands-free user interface for VR headsets based on in situ facial gesture sensing,” *Sensors*, vol. 20, no. 24, p. 7206, 2020.

[68] J. Cha, J. Kim, and S. Kim, “Hands-free user interface for AR/VR devices exploiting wearer’s facial gestures using unsupervised deep learning,” *Sensors*, vol. 19, no. 20, p. 4441, 2019.

Hindawi
Computational Intelligence and Neuroscience
Volume 2021, Article ID 4632353, 18 pages
https://doi.org/10.1155/2021/4632353

Research Article

Robust Real-Time Traffic Surveillance with Deep Learning

Jessica Fernández, José M. Cañas, Vanessa Fernández, and Sergio Paniego

Universidad Rey Juan Carlos, Móstoles, Spain

Correspondence should be addressed to José M. Cañas; josemaria.plaza@urjc.es

Received 23 July 2021; Accepted 10 November 2021; Published 27 December 2021

Academic Editor: Miguel Cazorla

Real-time vehicle monitoring in highways, roads, and streets may provide useful data both for infrastructure planning and for traffic management in general. Even though it is a classic research area in computer vision, advances in neural networks for object detection and classification, especially in the last years, made this area even more appealing due to the effectiveness of these methods. This study presents TrafficSensor, a system that employs deep learning techniques for automatic vehicle tracking and classification on highways using a calibrated and fixed camera. A new traffic image dataset was created to train the models, which includes real traffic images in poor lightning or weather conditions and low-resolution images. The proposed system consists mainly of two modules, first one responsible of vehicle detection and classification and a second one for vehicle tracking. For the first module, several neural models were tested and objectively compared, and finally, the YOLOv3 and YOLOv4-based network trained on the new traffic dataset were selected. The second module combines a simple spatial association algorithm with a more sophisticated KLT (Kanade–Lucas–Tomasi) tracker to follow the vehicles on the road. Several experiments have been conducted on challenging traffic videos in order to validate the system with real data. Experimental results show that the proposed system is able to successfully detect, track, and classify vehicles traveling on a highway on real time.

1. Introduction

Number of vehicles on earth is increasing rapidly. According to data provided by International Organization of Motor Vehicle Manufacturers (OICA, https://www.oica.net/), the number of vehicles produced in the last years is way more than 70 million vehicles per year. This number is increasing very quickly, equally the number of travel kilometers increases even more quickly. This explosion in the number of moving vehicles raises several challenges of different types: environmental, economical, and infrastructure management. At this moment, it is clear that managing such large number of vehicles is one of the biggest problems that countries worldwide have to deal with. Classic vehicle monitoring techniques cannot deal with such huge amount of data nor make an intelligent use of it. It is clear that new sophisticated paradigms are needed to deal with this challenging task.

The main goal of intelligent transportation systems (ITSs) is to monitor the different vehicle transport networks in a smart way. For this, they make use of the different available technologies such as dedicated sensors and advanced video cameras. The objective of this monitoring is to extract useful information that can be used to coordinate the vehicle traffic networks. Eventually, by means of these systems, we want to minimize congestion and to enhance mobility.

Video cameras are the most used sensors on ITSs systems. Their simple installation and maintenance combined with their rich nature of the information make them one of the best solutions when it comes to surveillance and monitoring. Depending on the conditions of the ITSs system, it will be necessary to use moving cameras or fixed cameras. In addition to the cameras, ITSs systems traditionally made use of other sensors such as radars for speed enforcement or inductive loops and laser and infrared sensors for vehicle classification [1–6]. Systems based on these sensors try to classify the vehicles by extracting certain information such as the vehicle's length and number or distance between axles. Although they may provide a better accuracy in general, they require an intrusive installation and not all of them provide the possibility of multilane monitoring. Another drawback is their initial high installation cost, which is an important factor to take into account when comparing ITS systems. In fact, when evaluating this kind of systems, not only the initial price but also the whole system life cycle must be taken into consideration. Finally, the information provided by these traditional systems is basic and cannot be used to extract high level traffic data such as vehicle orientation, position, or other parameters that can be used for traffic law enforcement.

Usage of video cameras in traffic surveillance [7–9] typically was limited to passive monitoring tasks or very basic automated processing. The advances in image processing algorithms in the last decade specially in the deep neural networks area have opened the door to more sophisticated systems based on computer vision. Nowadays, with these advances, we can create systems not only able to detect vehicles in normal situations but with capacity to recognize and classify vehicles in very challenging situations. This may be the base to perform high level tasks such as automated traffic management, automatic incident detection, law enforcement, fog, and other weather conditions and many other incidents.

The study presents a vision-based traffic monitoring system, named TrafficSensor, that includes a robust vehicle detection and classification algorithm and a new technique for dealing with occlusions [10–12]. It is the evolution of a previous system [13] towards a higher reliability and good performance even in challenging lightning or weather conditions, and poor camera resolution while keeping real-time operation. TrafficSensor is based on the use of a fixed camera to detect and monitor vehicles. Section 2 (Related Works) reviews relevant studies on vehicle classification [14–18]. The system core functionality is described in the Section 3 (TrafficSensor: A Deep Learning-Based Traffic Monitoring Tool), where the details for the vehicle tracking and deep learning-based detection algorithms are presented. Section 4 (Experimental Validation) presents several tests performed to validate the system functionality and the quantitative obtained results. Finally, Section 5 (Conclusion) summarizes the main lessons extracted from this work.

2. Related Works

The literature provides many publications dealing with vehicle monitoring [19–21], even recognizing the vehicle model [22]. To perform such monitoring, it is necessary to detect the vehicles and then to follow them up. A technique widely used for vehicle detection is background subtraction [23–27]. The background subtraction technique is a technique widely used to detect objects such as the difference between a current pixel and a reference pixel, called background. Huang [28] used the Gaussian mixture to detect the background and subsequently subtract it. This guarantees that the background we extract corresponds to the lighting of that moment [29]. The mixture of Gaussian (MOG) proposes to model the intensity of the pixels with a mixture of k Gaussian distributions. MOG is a technique that first applied to the problem of background subtraction. TrafficMonitor [13] makes use of an improved version of the proposed MOG by Zivkovic [30]. The advantage of this method is that for each pixel, the number of Gaussian to be used can be adapted. Another technique very similar to background subtraction is the absolute difference (sum of absolute differences (SAD)) between two sequences. Samhitha et al. [31] presented a technique based on the absolute difference (SAD) between two consecutive frames. Guerrero-Gomez-Olmedo et al. [32] used the histogram of oriented gradients (HOG) to detect vehicles. HOG is a type of feature descriptor. It converts the local information of the gradients for each pixel into a representation of the image that captures the global shape of the object into a feature vector.

For vehicle tracking [33–38], many solutions rely on the features. To follow-up, Wang et al. [39] employed a technique based on features called scale-invariant feature transform (SIFT) [34] and optical flow. SIFT is an algorithm used to extract characteristics from images. Optical flow is the pattern of movement of the image objects between two consecutive frames caused by the movement of the object. Mu et al. [35] also used SIFT to track vehicles. Huang and Barth [40] proposed an algorithm to carry out vehicle tracking and resolution of occlusions. In this algorithm, they use a color model based on mean-shift to identify which vehicle each 3×3 pixel patch belongs to when there is an occlusion. In other cases, 2D or 3D [41] models are used to do the tracking. Leotta and Mundy [42] employed this technique to detect vehicles using a deformable template that adjusts to identify different forms of vehicles. Huang [28] and Baker and Sullivan [43] used Kalman filters and Guerrero-Gómez-Olmedo et al. [32] employed extended Kalman filters (EKF) [44]. The Kalman filter is an algorithm to update, observation by observation, the linear projection of a system of variables on the set of available information, as new information becomes available. The extended Kalman filter consists of a variation of the Kalman filter to trackle the state estimation problem when the model is possibly nonlinear.

Regarding image classifiers, Vedaldi et al. [45] proposed a novel three-stage classifier, which combines linear, quasilinear, and nonlinear kernel SVMs. They showed that increasing the nonlinearity of the kernels increases their discriminative power, at the cost of an increased computational complexity. Their aim was to learn an SVM classifier [46], where rather than using a prespecified kernel, the kernel is learnt to be a linear combination of given base kernels.

One of the most heavily studied paradigms for object detection [47, 48] and classification is deep learning. The convolutional neural network (CNN) is a feed-forward type of the machine learning algorithm that have shown impressive results and robustness in visual object detection. They have been widely explored in the context of vehicle monitoring too. In Migel et al.'s work [49], the vehicle identification and classification are performed for each extracted portion of the input image, simultaneously using the designed CNN. That is, a softmax layer is used as the classifier to perform vehicle classification. Caffe [50] framework was used to benchmark the performance of the vehicle detection system. Sensa et al. [51] presented an intelligent traffic congestion detection method using the CNN. The dataset used in this experiment is the road traffic condition images from CCTV camera in Jakarta during 29 April–5 May 2017 that can be obtained from lewatmana.com http://lewatmana.com.

Yang et al. [52] proposed a detection method using a single image to generate the 3D space coordinate information of the object using monocular vision for autonomous driving. Their method is built by modifying the fast R-CNN using multitask learning, and thus is named multitask faster R-CNN (MT faster RCNN). For the experiments, the KITTI dataset was used.

Luo et al. [53] presented a model based on the faster RCNN with NAS optimization and feature enrichment to perform the effective detection of multiscale vehicle targets in traffic scenes. Luo et al. proposed a Retinex-based image adaptive correction algorithm (RIAC) (to reduce the influence of shadows and illumination), conducted neural architecture search (NAS) on the backbone network used for feature extraction of the faster RCNN (to generate the optimal cross-layer connection to extract multilayer features more effectively), and used the object feature enrichment that combines the multilayer feature information and the context information of the last layer after cross-layer connection (to enrich the information of vehicle targets and improve the robustness of the model for challenging targets such as small scale and severe occlusion). Their model has been trained and tested on the UN-DETRAC dataset.

Redmon et al. [54] presented YOLO, a new approach to object detection. YOLO reframes object detection as a single regression problem, straight from image pixels to bounding box coordinates and class probabilities. Use You Only Look Once (YOLO) at an image to predict what objects are present and where they are. Jean-Francois Rajotte et al. [55] did automatic annotations that were performed with the YOLO detector. Kwan et al. [56] used YOLOv1 to detect vehicle in real time.

Mahto et al. [57] used the object detection algorithm YOLOv4 and optimized it for vehicle detection. To improve YOLOv4, they proposed optimize the anchor box using k-means clustering (ABK), the nonmaximum suppression with distance-IoU (DIoU-NMS), the spatial attention module (Sam), and the self-adversarial training (SAT). The UA-DETRAC Benchmark dataset was used to train and test the method.

Zhang et al. [58] proposed an improved RetinaNet. Their algorithm uses octave convolution instead of the traditional convolution layer and a weighted feature pyramid network (WFPN) structure to limit the propagation of gradients between different levels. To evaluate the result, the DETRAC dataset was used.

Szegedy et al. [59] presented a network that is based on the convolutional DNN defined by [60]. It consists of total 7 layers, the first 5 of which being convolutional and the last 2 fully connected. Each layer uses a rectified linear unit as a nonlinear transformation. Three of the convolutional layers have in addition max pooling.

3. TrafficSensor: A Deep Learning-Based Traffic Monitoring Tool

TrafficSensor tool is able to monitor traffic in real time and classify the vehicles into 7 categories: motorcycles, cars, vans, buses, trucks, small trucks, and tank trucks. It consists of three main blocks: vehicles detection, vehicles classification, and vehicles tracking, as shown in Figure 1. They are implemented in two separate modules, as detections and their classification are carried out jointly because deep learning is used. The tracking focuses on spatial proximity, and if it fails, KLT is used. All detected blobs will be tracked over time.

There is a single image area where detection, classification, and tracking are carried out. This area, that is called evaluation area, is marked in the image by the user to identify where on the road we want to focus the detections, as shown in Figure 2. TrafficSensor is designed to monitor outgoing traffic flow, although it can be extrapolated to incoming traffic flow.

3.1. Deep Learning-Based Detection and Classification.

The system takes input images acquired from the video being monitored. These images pass as input to the neural network, where various vehicles are detected and classified. All information is stored at each moment, so that it can be tracked based on the information recorded from the previous moment. TrafficSensor supports trained neural networks with different neural frameworks (TensorFlow, Darknet, and Keras) in order to detect and classify the different vehicles that appear in the image.

In the detection and classification block, the system implements these criteria:

(i) Inside the evaluation area, there are two zones (Figure 3). The zone 1 matches with the half of the evaluation area where the vehicles enter. In this zone, it is easier to detect and classify the vehicles because they are bigger than that in other areas of the image. The zone 2 refers to the half through which vehicles leave the evaluation zone. This zone is more complex, since the vehicles have smaller size than in the zone 1.

(ii) Vehicles always enter in the evaluation area through zone 1. They can never appear suddenly. For this reason, no new vehicle can appear in the middle of the road. A new vehicle can never be detected in zone 2.

(iii) If a vehicle is not detected in zone 1 during five-frame sequence, it will be a false positive. It will be discarded.

(iv) Any vehicle that is in zone 2 will be considered a correct vehicle. If the vehicle is not detected with deep learning, KLT will be used to locate it.

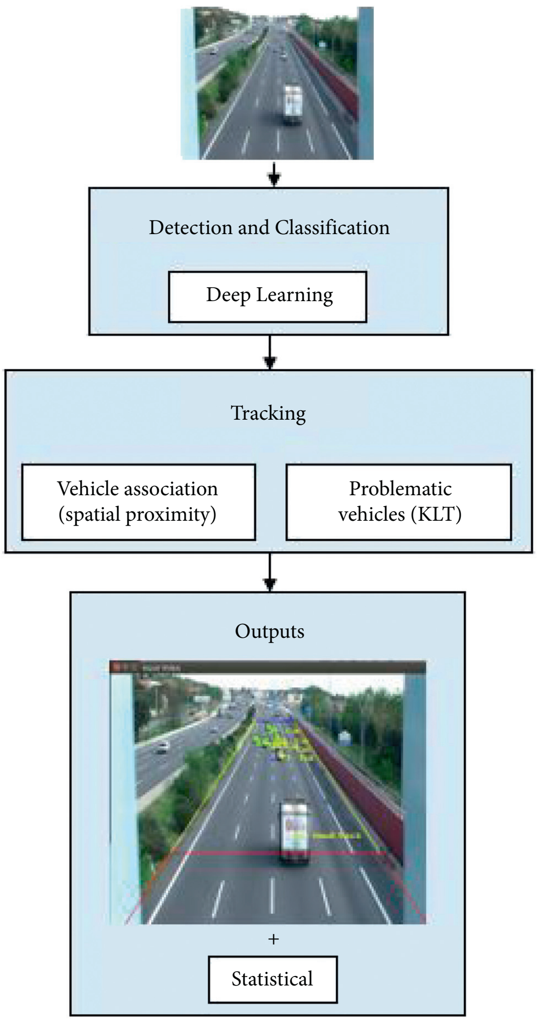

Detection and Classification
Deep Learning
Tracking
Vehicle association (spatial proximity)
Problematic vehicles (KLT)
Outputs
+
Statistical

Figure 1: Block diagram of the TrafficSensor system.

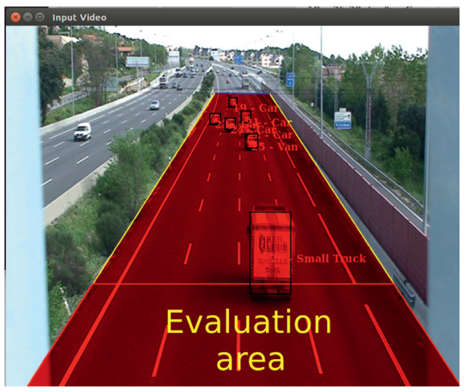


Figure 2: Evaluation area.

Three different frameworks (TensorFlow, Keras, and Darknet) and four neural network models have been tested in order to evaluate which one is better for the final TrafficSensor application. Specifically, the SSD MobileNetV2 network with TensorFlow, the SSD VGG-16 network with Keras, and YOLOv3 and YOLOv4 with Darknet.

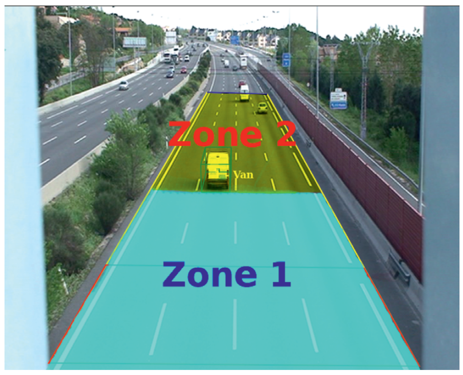


Figure 3: Evaluation zones.

3.1.1. SSD MobileNetV2 Network. The SSD MobileNetV2 network (Figure 4) used was trained with the COCO dataset. To use this network, we utilized the configuration file ssd mobilenet v2 coco.config.

This network is formed by a SSD and a MobileNet V2. MobileNet V2 gets the maps of features to perform the classification and detection in the subsequent layers. The SSD approach is based on a convolutional feed-forward network that produces a set of bounding boxes fixed in size and punctuates the presence of object class instances in those bounding boxes. After this, it carries out nonmaximum suppression to produce the final detections.

3.1.2. SSD VGG-16 Network. Another SSD network has been used with VGG-16 as its base network, pretrained with ImageNet. Figure 5 shows this network model. VGG-16 consists of 16 layers, of which 13 are convolutional layers, 2 fully connected layers, and a softmax layer that is used to classify. Figure 6 shows how the architecture of the VGG-16 network is.

3.1.3. YOLOv3. You Only Look Once (YOLO) imposes strong spatial constraints on bounding box predictions, since each cell in the grid only predicts N bounding boxes (N being a fixed parameter) and can only have one class. This spatial restriction limits the number of nearby objects that our model can predict.

The YOLOv3 [61] network (Figure 7) is made up of a total of 107 layers, which can be grouped into two groups, one in charge of extracting features and another in charge of detecting objects:

(i) Feature extraction (from layers 1 to 75): it is the Darknet-53 network trained with ImageNet, which is composed of 53 convolutional layers (Figure 8). This network has $416 \times 416 \times 3$ images as input and features as output 3D $13 \times 13 \times 1024$ and incorporates 23 residual layers. When a neural network increases in depth its precision when it comes to propagating the characteristics, it tends to degrade, leading to a greater error in training. The residual layers are used to solve this problem.

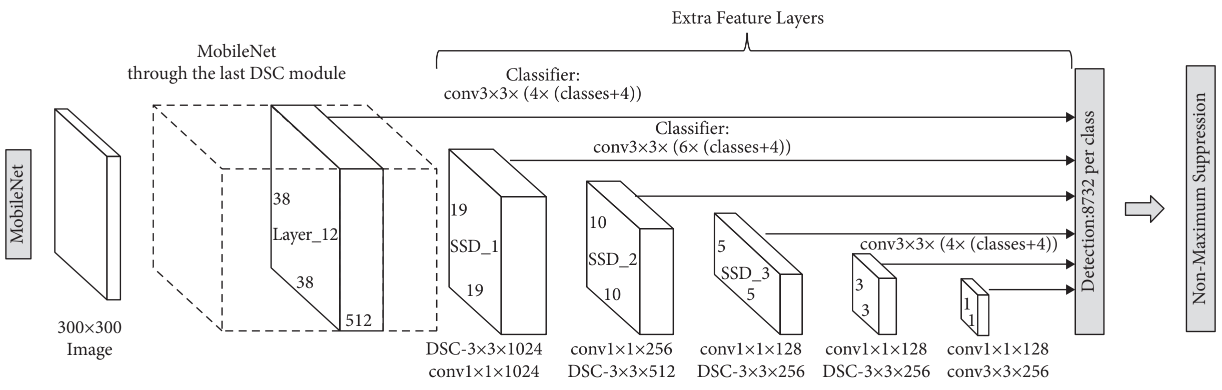


Figure 4: SSD MobileNet V2 network.

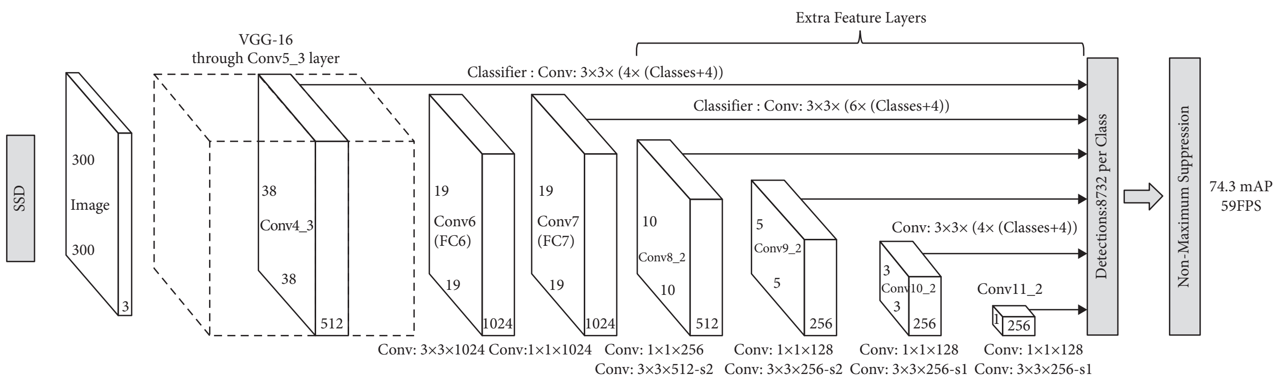


Figure 5: SSD network model.

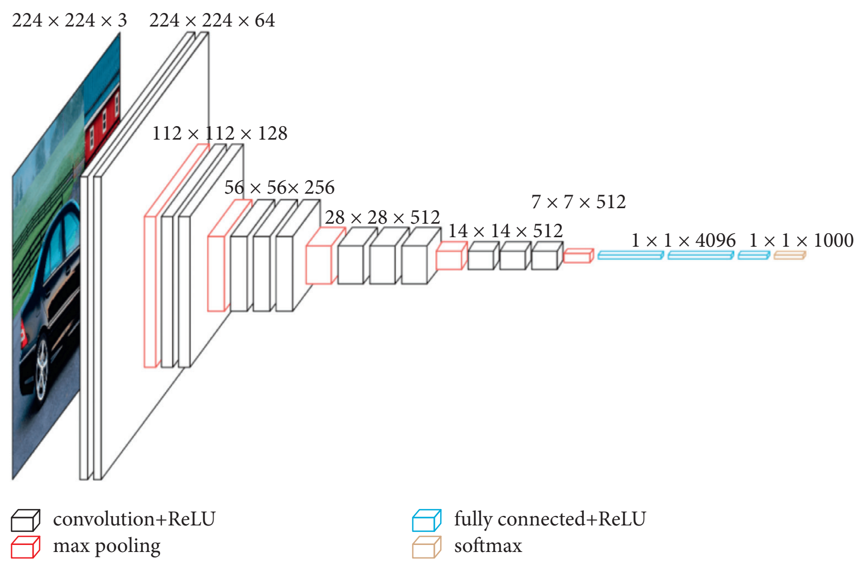


Figure 6: VGG-16 model.

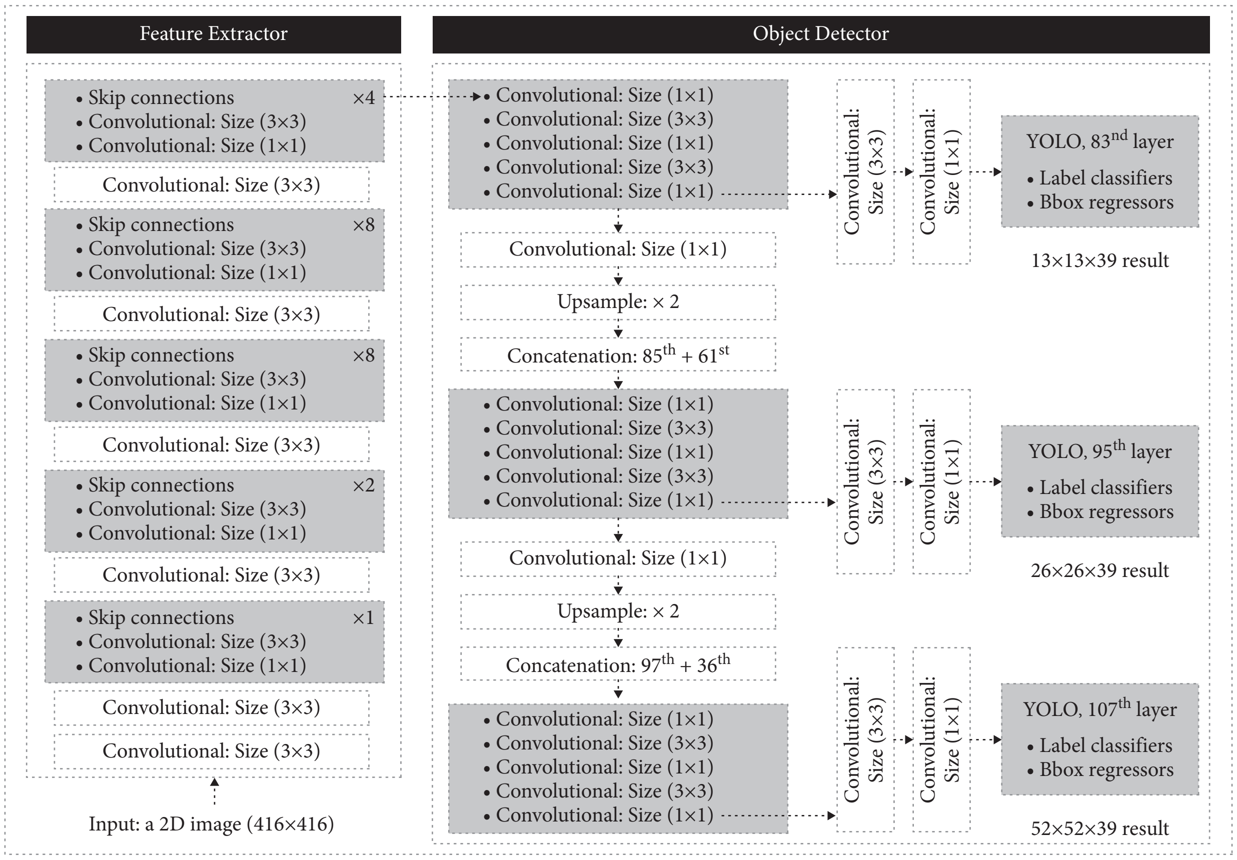
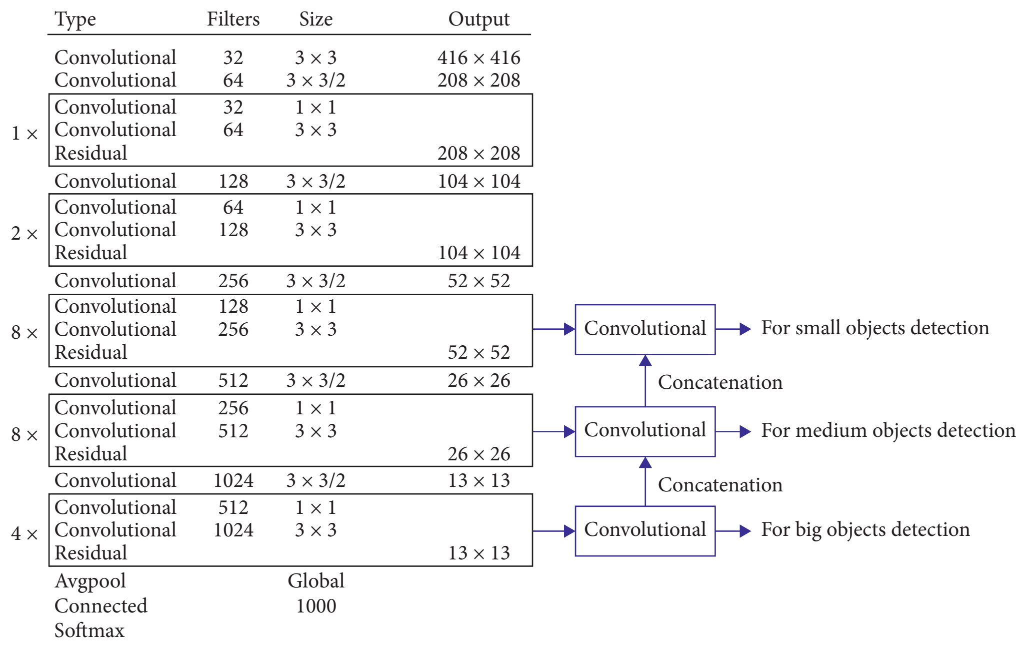

Feature Extractor
• Skip connections ×4
• Convolutional: Size (3×3)
• Convolutional: Size (1×1)
Convolutional: Size (3×3)
• Skip connections ×8
• Convolutional: Size (3×3)
• Convolutional: Size (1×1)
Convolutional: Size (3×3)
• Skip connections ×8
• Convolutional: Size (3×3)
• Convolutional: Size (1×1)
Convolutional: Size (3×3)
• Skip connections ×2
• Convolutional: Size (3×3)
• Convolutional: Size (1×1)
Convolutional: Size (3×3)
• Skip connections ×1
• Convolutional: Size (3×3)
• Convolutional: Size (1×1)
Convolutional: Size (3×3)
Convolutional: Size (3×3)
Input: a 2D image (416×416)
Object Detector
• Convolutional: Size (1×1)
• Convolutional: Size (3×3)
• Convolutional: Size (1×1)
• Convolutional: Size (3×3)
• Convolutional: Size (1×1)
Convolutional: Size (3×3)
Convolutional: Size (1×1)
YOLO, 83nd layer
• Label classifiers
• Bbox regressors
13×13×39 result
Convolutional: Size (1×1)
Upsample: × 2
Concatenation: 85th + 61st
• Convolutional: Size (1×1)
• Convolutional: Size (3×3)
• Convolutional: Size (1×1)
• Convolutional: Size (3×3)
• Convolutional: Size (1×1)
Convolutional: Size (3×3)
Convolutional: Size (1×1)
YOLO, 95th layer
• Label classifiers
• Bbox regressors
26×26×39 result
Convolutional: Size (1×1)
Upsample: × 2
Concatenation: 97th + 36th
• Convolutional: Size (1×1)
• Convolutional: Size (3×3)
• Convolutional: Size (1×1)
• Convolutional: Size (3×3)
• Convolutional: Size (1×1)
Convolutional: Size (3×3)
Convolutional: Size (1×1)
YOLO, 107th layer
• Label classifiers
• Bbox regressors
52×52×39 result

Figure 7: YOLOv3 model.


Convolutional
For small objects detection
Concatenation
Convolutional
For medium objects detection
Concatenation
Convolutional
For big objects detection

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	416 × 416
	Convolutional	64	3 × 3/2	208 × 208
1 ×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			208 × 208
	Convolutional	128	3 × 3/2	104 × 104
2 ×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			104 × 104
	Convolutional	256	3 × 3/2	52 × 52
8 ×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			52 × 52
	Convolutional	512	3 × 3/2	26 × 26
8 ×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			26 × 26
	Convolutional	1024	3 × 3/2	13 × 13
4 ×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			13 × 13
	Avgpool		Global	
	Connected		1000	
	Softmax			

Figure 8: Darknet-53 model.

(ii) Objects detection (from layers 76 to 107): it takes the 3D features ($13 \times 13 \times 1024$) as input and with that performs object detection. The uniqueness of this network lies in its ability to detect objects on three different scales, making it a very powerful network before the change of scale. To do this, it extracts characteristics on three different scales ($13 \times 13 \times 39$, $26 \times 26 \times 39$, and $52 \times 52 \times 39$). These characteristics pass to the final YOLO layer, which classifies the object label with class logistic regressions and locates objects with bounding boxes regressors.

3.1.4. YOLOv4. YOLOv4 [62] is the fourth iteration of the famous YOLO architecture that continues improving the previous versions with the latest advances introduced in the literature. It consists of 3 main components: backbone, neck, and head (Figure 9). For the backbone, it uses CSPDarknet53 [63], for the neck, SSP [64] and PAN [65], and YOLOv3 [61] for the head.

This network allows real-time object detection on a conventional GPU, thanks to its improvements on speed comparing it to other approaches and even its previous versions.

3.2. Vehicle Tracking. The execution flow of the tracking module is shown in Figure 10, which shows the steps when a new blob is detected inside the image, and in Figure 11, which illustrates the procedure that is carried out on already registered vehicles. The tracking focuses on associating the current detections with the vehicles stored at the previous instant. Several points are considered:

(i) If a vehicle arrives at the end of the evaluation area, it will be removed from tracking
(ii) The vehicles stored of the instant $(t-1)$ are examined in order to pair them with the vehicles detected at the instant (t). This pairing will be carried out between the vehicles in (t) and $(t-1)$, which have the least Euclidean distance between their centers.
(iii) If the vehicle t associated with the vehicle $(t-1)$ is not within the circular or elliptical area around the center of the vehicle $(t-1)$, it will not be matched to it
(iv) If through space proximity we are not able to pair a $(t-1)$ vehicle, we will use KLT

Spatial proximity and KLT algorithm are used to perform the vehicle tracking. Spatial proximity works fine for separate vehicles, but in real videos, it is very likely that we have occlusions and vehicles that are quite complex to detect, especially when they are small in the image because they are far from the camera. The feature-based tracking algorithm KLT is used then, and it is a good complement to the deep learning robustness.

3.2.1. Spatial Proximity Tracking. Typically, the difference of pixels in the image between the position of a vehicle in $(t-1)$ and in (t) is very small. Therefore, a vehicle in (t) will be in an area very close to that same vehicle in $(t-1)$. When we search for a vehicle in (t), we should find it in a small circular radius around the position of that same vehicle in $(t-1)$. Spatial proximity tracking in TrafficSensor is based on [13]. It estimates the area where you should locate a vehicle based on its position in $(t-1)$. As the vehicles move forward, this area will be updated.

At first, the area is taken as a circle because the system does not have enough data about its orientation. But as the vehicle advances, the system has enough information to know its orientation, and so, it takes the area as an ellipse whose center corresponds to the center of the vehicle in $(t-1)$. It is considered that we have enough information to estimate its orientation when we have the position of the vehicle in 6 frames. Linear regression is used to calculate the orientation of the vehicle based on the position the vehicle will take as it progresses. Once we have information about the orientation, we will define the search area as an ellipse whose center is the same as the vehicle in $t-1$ and direction calculated with the following equation.

$$\rho(r_i) = 2\left(\sqrt{1+\frac{r^2}{2}}-1\right). \quad (1)$$

The pairings between the vehicles detected at time (t) and the vehicles stored from time $(t-1)$ are limited to vehicles that fall within the area of the circle or the ellipse that is obtained based on the position of the vehicle at time $(t-1)$. The ellipses are defined as $C_{xc,yc,\omega}$, where ω is the orientation and (x_c, y_c) is the center of the vehicle. These parameters are shown in Figure 12.

The 2D vehicle, whose center is $B(x, y)$, will be inside the ellipse $C_{xc,yc,\omega}$, if it accomplishes the following equations:

$$C_\omega = \arctan\left(\frac{a_x}{a_y}\right), \quad (2)$$

$$\left(\frac{\cos(C_\omega)(B_x - C_{x_c}) + \sin(C_\omega)(B_y - C_{y_c})}{b}\right)^2 + \left(\frac{\cos(C_\omega)(B_y - C_{y_c}) - \sin(C_\omega)(B_x - C_{x_c})}{a}\right)^2 \leq 1, \quad (3)$$

where a_x and a_y are the components of the orientation vector. Figure 13 shows the tracking between two consecutive vehicles.

Figure 14 shows an example of TrafficSensor where the tracking of two vehicles by space proximity is shown. The vehicle identified as 2 in the image of the instant $(t-1)$ is associated with the closest vehicle to its position in the current image (t).

A detection must be within a certain area around the blob detected in $(t-1)$ to be identified as the same vehicle. It could happen in the case that two vehicles will fall into that area. Therefore, it is necessary to take into account the Euclidean distance between the center of the blob of the instant $(t-1)$ and the center of the blob in (t). The blob of the instant (t) that is at a smaller distance from the blob of the instant $(t-1)$ and of course within the area around blob $(t-1)$ will be considered the same vehicle than that of $(t-1)$. That is to say, if this is true, the blobs of $(t-1)$ and (t) correspond to the same vehicle but in consecutive moments.

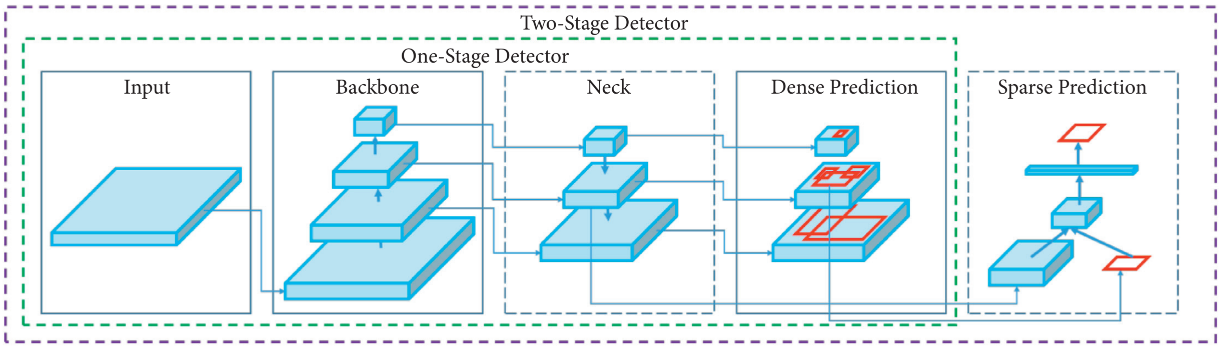


Figure 9: YOLOv4 object detector.

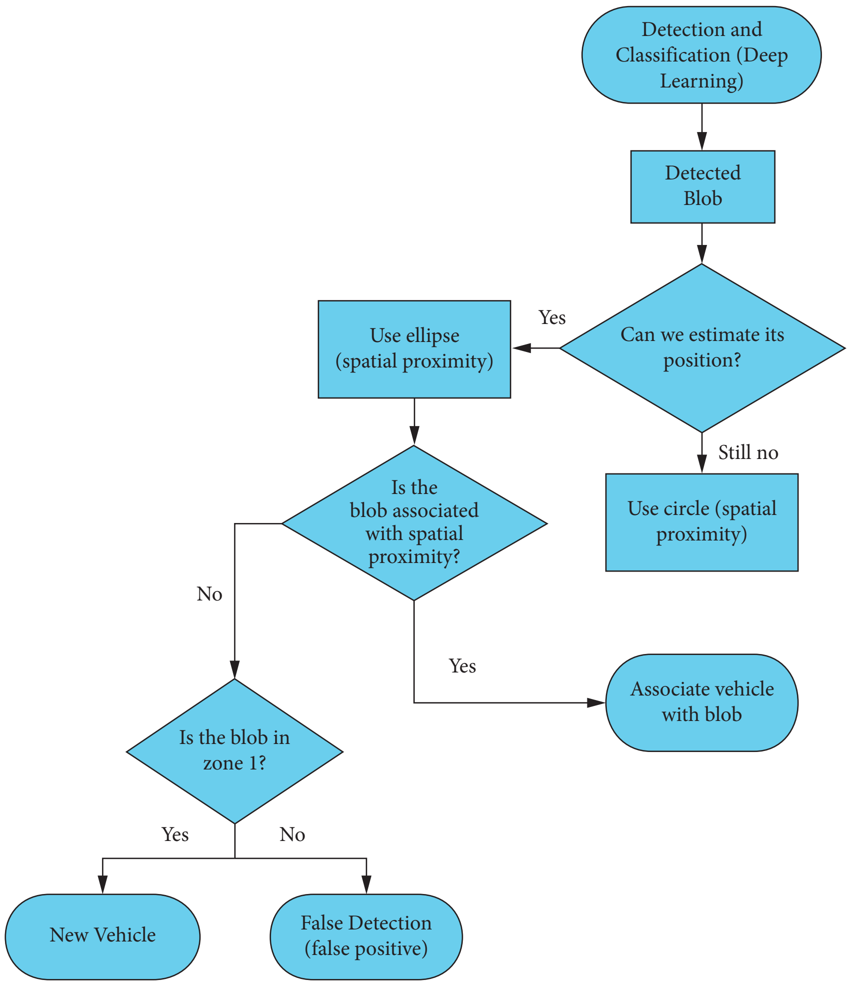

Detection and Classification (Deep Learning)
Detected Blob
Can we estimate its position?
Yes
Use ellipse (spatial proximity)
Still no
Use circle (spatial proximity)
Is the blob associated with spatial proximity?
No
Yes
Associate vehicle with blob
Is the blob in zone 1?
Yes
No
New Vehicle
False Detection (false positive)

Figure 10: Execution flowchart of detected blobs.

3.2.2. KLT Tracking. The follow-up is mainly based on spatial proximity, but KLT will be used in problematic cases, thus making our system more robust. KLT will be calculated in all sequences to update the feature points. If a vehicle is not detected either because there is an occlusion or it is very far away, KLT will be used, as it has proven to work well even in occlusions during a small number of consecutive frames.

To use KLT, we need to know the center of mass of the vehicles and their visual features. Depending on the feature points of the vehicle in $(t-1)$, KLT calculates the matching for each feature point and as a result generates a new set of feature points corresponding to the vehicle in question. In order to achieve a correct match, the system is based on votes of the feature points that an object has associated. Figure 15 shows an example.

KLT is a feature tracking algorithm [? ?]. KLT is a differential and local method in which the neighborhood of each pixel is analyzed. The algorithm assumes that the

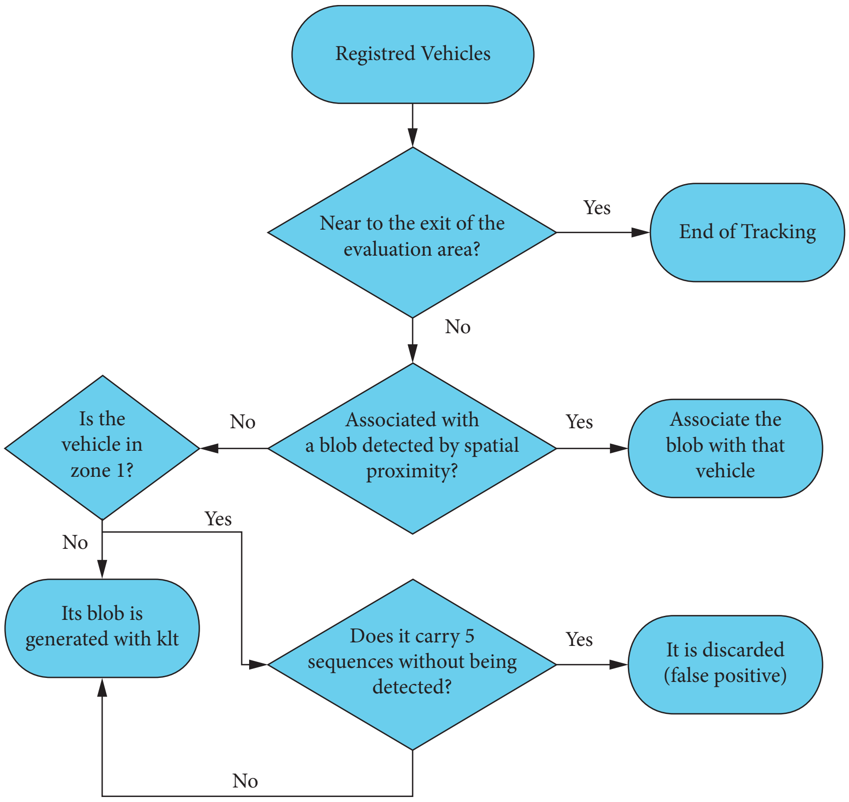

Registred Vehicles
Near to the exit of the evaluation area?
Yes
End of Tracking
No
Associated with a blob detected by spatial proximity?
Yes
Associate the blob with that vehicle
No
Is the vehicle in zone 1?
Yes
No
Its blob is generated with klt
Does it carry 5 sequences without being detected?
Yes
It is discarded (false positive)
No

Figure 11: Flowchart of registered vehicles.

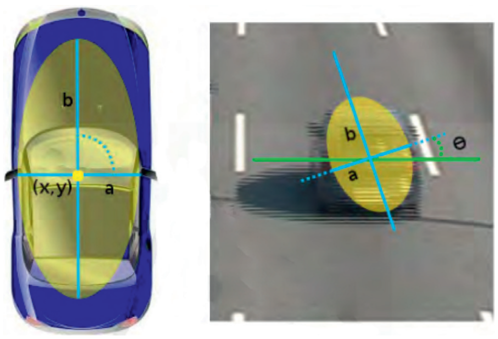


Figure 12: Vehicle-associated 2D ellipse.

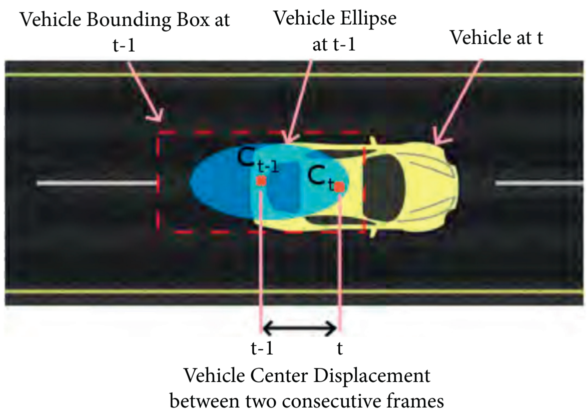

Vehicle Bounding Box at t-1
Vehicle Ellipse at t-1
Vehicle at t
C_{t-1}
C_t
t-1
t
Vehicle Center Displacement between two consecutive frames

Figure 13: Proximity tracking ellipse.

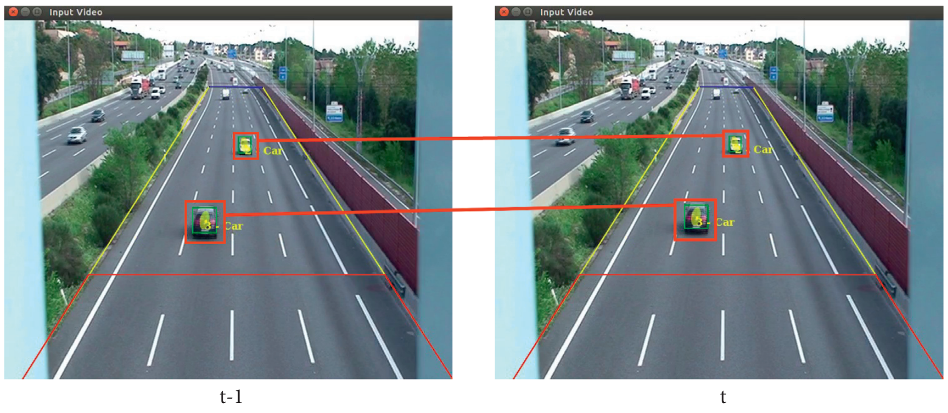


Figure 14: Tracking with spatial proximity TrafficSensor.

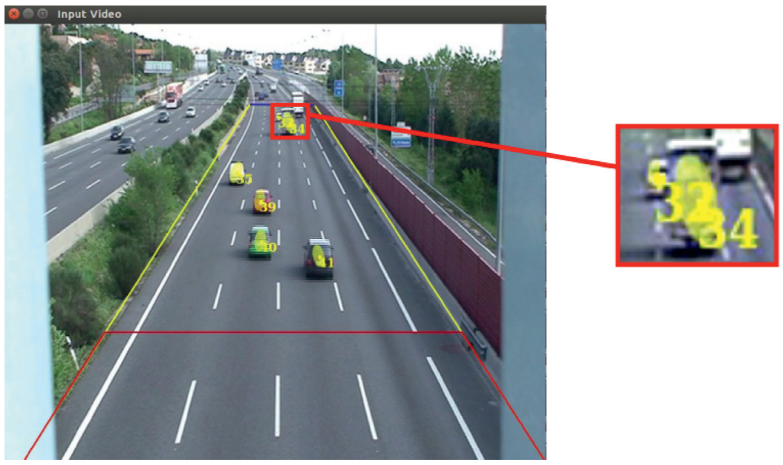


Figure 15: Tracking with KLT in TrafficSensor.

optical flow is constant in a neighborhood. The equation of the optical flow is solved for all the pixels in this neighborhood by the method of least squares. For the calculation of the velocity vectors, the following formula is used:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum_i I_{xi}^2 & \sum_i I_{xi} I_{yi} \\ \sum_i I_{xi} I_{yi} & \sum_i I_{yi}^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i I_{xi} I_{ti} \\ -\sum_i I_{yi} I_{ti} \end{bmatrix}. \quad (4)$$

The vector (u, v) is the displacement vector of the optical flow. I_x is the mean of gradient in x between two consecutive images, that is, if $I(t)$ is the image of the instant current and $I(t+1)$ is the image at the next instant, I_x of these frames is

$$I_x = \frac{I_x(t) + I_x(t+1)}{2}, \quad (5)$$

where $I_x(t)$ is the gradient in the x axis of the image $I(t)$ and $I_x(t+1)$ is the gradient in x of the image $I(t+1)$. I_y is the mean of the gradients in y of the image $I(t)$ and $I(t+1)$:

$$I_y = \frac{I_y(t) + I_y(t+1)}{2}. \quad (6)$$

It is the difference between $I(t)$ smoothed and $I(t+1)$ smoothed:

$$I_t = I'(t+1) - I'(t). \quad (7)$$

KLT is applied in the form of kernels of size $\omega \times \omega$ throughout from image. The size of the kernels must be defined according to the amount of movement that the image has. A small kernel would be ideal for evaluating small displacements of a point. Using a large kernel increases the risk of getting an error, but there are cases where the displacement of a point is very big and this is necessary.

TrafficSensor uses the pyramidal implementation [66], which Jean-Yves Bouguet introduced. On it, the KLT algorithm is applied recursively over an image pyramid, as shown in Figure 16.

4. Experimental Validation

The proposed system has been validated with a dataset of real traffic images, which has been divided into training and test subsets. In addition, the 4 studied neural networks for the detection and classification module of TrafficSensor have

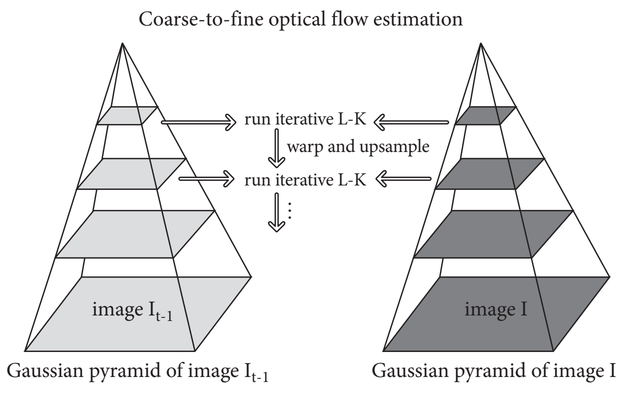

Coarse-to-fine optical flow estimation
run iterative L-K
warp and upsample
run iterative L-K
image I_{t-1}
image I
Gaussian pyramid of image I_{t-1}
Gaussian pyramid of image I

Figure 16: Pyramidal KLT.

been quantitatively compared using an open source tool, named DetectionMetrics, so the best one could be selected for the final system. This measuring tool is publicly available and was created as a part of this work, but it is generic and usable in any other visual detection application. In addition, the final system was tested and validated both with good lightning images and, in particular, with poor images or images in bad weather conditions, which are typically present in real deployments.

4.1. Dataset. To train and evaluate the networks, a new dataset was created. This dataset includes images in good weather conditions, images in bad weather conditions (with fog and rain), and poor quality images. This dataset consists of the following:

(i) The database built by Redouane Kachach in his doctoral thesis [13]. That database consists of 3460 good quality images.

(ii) The GRAM Road-Traffic Monitoring (GRAM-RTM) database created by Guerrero-Gomez-Olmedo et al. [32]. This database is made up of images extracted from three videos. The first video, called M-30 (7520 frames), was recorded on a sunny day. The second, called M-30-HD (9390 frames), was recorded in a similar location but during a cloudy day. The third, called Urban1 (23435 frames), was recorded at a busy intersection. From this large database, 3646 images of the M-30-HD video and 1348 of the M-30 video were used.

(iii) Images were collected from open online cameras. 615 were about rain situations and 705 of poor quality images.

In total, the dataset for TrafficSensor consists of 9774 images. All of them have been manually tagged with the labelImg tool https://github.com/tzutalin/labelImg, using 7 possible classes: car, motorcycle, van, bus, truck, small truck, and tank truck. In these 9774 images, we have a total of 48914 samples distributed, as given in Table 1.

Table 2 provides the number of images that exist for each type of image (good conditions, bad weather, and poor quality), and Figure 17 shows some illustrative images of our database.

Table 1: Database samples.

Class	Sample
Car	38976
Motorcycle	1886
Van	5631
Bus	401
Truck	963
Small truck	938
Tank truck	119

Table 2: Database images.

	N of images
Good conditions	8406
Bad weather	663
Poor quality	705

Of these 9774 images, one part was used in training and another in the test. Table 3 provides the distribution of images according to training and test.

For the training of the involved neural networks, the training database was divided itself into train and validation subsets. Out of the 9246 images, 7401 were used as train and 1845 as validation. Table 4 provides the number of images that has been used in training depending on its type (good quality, poor quality, and bad weather).

4.2. DetectionMetrics Tool. DetectionMetrics (https://jderobot.github.io/DetectionMetrics/) is an opensource research software application that has been created and used to quantitatively evaluate the performance of pretrained neural networks and our visual traffic surveillance application.

It provides a toolbox of utilities oriented to simplify the development and testing of solutions based on visual object detection. The application comes with a GUI (based on Qt) and can also be used through command line. It is designed to generate experiment results from running a set of neural networks models over many datasets. Currently, it comes with the following utilities: viewer, detector, evaluator, deployer, labelling, and converter.

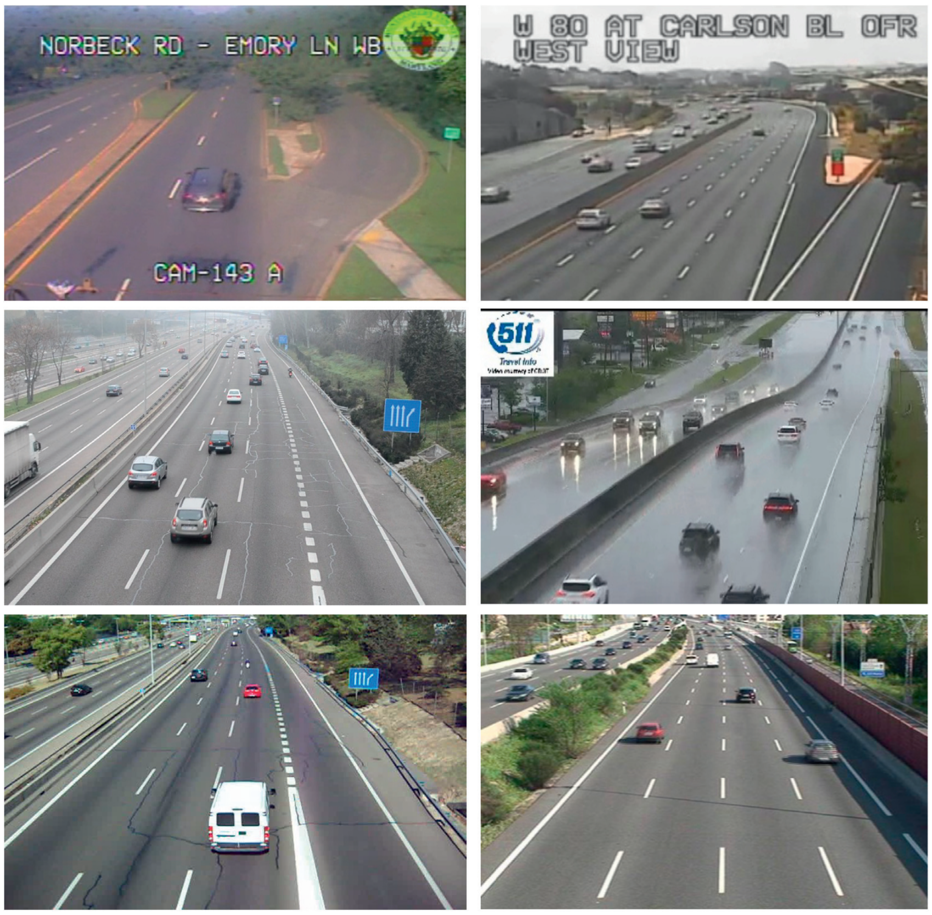


Figure 17: TrafficSensor dataset samples.

Table 3: Dataset distribution.

Type	Training images	Test images
Good conditions	6717	389
Bad weather	1892	71
Poor quality	637	68
Total	9246	528

Table 4: Training dataset.

Type	Training images	Validation images	Total
Good conditions	5323	1394	6717
Bad weather	1568	324	1892
Poor quality	510	127	637
Total	7401	1845	9246

It comprises a generic infrastructure to evaluate object detection algorithms against a dataset and calculate common statistics:

(i) IntersectionOverUnion (IoU) measures the accuracy of a detection in a particular dataset and follows the following formula:

$$\mathrm{IoU} = \frac{\mathrm{AreaofOverlap}}{\mathrm{AreaofUnion}}. \quad (8)$$

Here, AreaofOverlap is the area that belongs to the intersection between prediction and ground truth, while AreaofUnion is the sum area (without repetition of the overlap) of the prediction and ground truth as shown in Figure 18.

(ii) Precision is the total correct detections among the number of detections obtained. The precision of DetectionMetrics is the average (mean average precision (mAP)) for those predictions that have an IoU greater than a threshold (0.5).

$$\mathrm{Precision} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}. \quad (9)$$

(iii) Recall is the number of correct detections among the number of actual detections, that is, ground truth detections. Like precision, averaging (mean average recall (mAR)) of detections having a higher IoU is obtained to 0.5.

This tool is compatible with Linux, Windows, and MacOS because it is provided as a Docker image in addition to the common source code installation. It allows to evaluate models trained in TensorFlow, PyTorch, Keras, Caffe, and Darknet, and it supports the most common dataset formats in object detection (YOLO, COCO, ImageNet, and Pascal VOC) and can use different image input sources (videos and webcam).

The main workflow used for the experiments is called headless evaluation. This workflow involves mainly two of the tools included in DetectionMetrics: detector and evaluator. In this mode, a researcher determines a set of experiments that will run independently and unattended, retrieving a final report with the previous described objective metrics. DetectionMetrics receives a batch of datasets and deep learning pretrained models, predicts the objects on the images using each model over each dataset, and outputs the report with metrics of the performance for each scenario.

4.2.1. Detector. Detector generates a new annotated dataset with the predicted labels given a pretrained neural network model and a dataset. This new generated dataset contains the images from the datasets along with the detected objects, their position, and the level of confidence for the predictions. It supports the most common deep learning frameworks: TensorFlow, Keras, Darknet, Caffe, and PyTorch.

During the detection process, DetectionMetrics shows the predictions using viewer, showing the ground truth and predictions on the image while running. This is very convenient as qualitative feedback about the network performance.

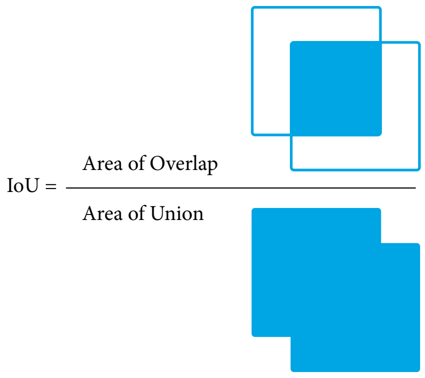


Figure 18: IoU formula.

4.2.2. Evaluator. Evaluator receives two datasets with the same format as input, one considered to be the ground truth and the other the generated detections dataset, and retrieves an evaluation report with metrics for every experiment, showing how each network was performed over each dataset. It supports both mAP and mAR metrics.

4.3. Comparison of Neural Networks. The four trained networks with 3 different neural frameworks were tested (SSD MobilenetV2, SSD VGG-16, and YOLOv3 and YOLOv4) with the images of good conditions. YOLOv3 and YOLOv4 have also been tested with the weights prior to training with our database. This experiment was performed on a GeForce RTX 3070 graphics card, whose main features are given in Table 5.

The quantitative results obtained in the experiment are given in Table 6. Those of the YOLO networks are better than those of SSD MobilenetV2 and SSD VGG-16. Looking at the detection times, it can be seen that all the trained networks show similar speeds. With the pretrained weights, the detection times are bigger because these weights have obtained training with more classes. The achieved quality results (mAP and mAR) with the pretrained weights are worse than with the trained weights as expected. This makes clear the need to retrain the network with an adequate database that adjusts the network model to the data we want to detect.

As given, YOLOv4 further improves both the detection quality and speed over YOLOv3. YOLOv4 uses data augmentation. It interprets the same information from different points of view. YOLOv4 is based on pixel-by-pixel modifications in the training images (color changes, texture, black or white patches, cuts, and other modifications) that help the algorithm to increase its precision and flexibility, but without affecting its performance in terms of speed. In the performance achieved in the tests, the speed of YOLOv4 versus YOLOv3 has increased by 13%. The result is practically equal to that indicated by the authors of YOLOv4 [62], who reported a speed increase of 12%.

Table 5: GeForce RTX 3070 specs.

GPU engine specs	
NVIDIA CUDA cores	5888
Base clock (GHz)	1.5
Boost clock (GHz)	1.73
Memory specs	
Memory speed	14 Gbps
Standard memory config	8 GB GDDR56
Memory interface width	256 bit
Memory bandwidth (GB/sec)	448

Table 6: Results of trained networks.

Neural networks	Framework	mAP	mAR	Mean inference time (ms)
ssd300adam.h5	Keras	0.7478	0.7831	13
ssd_mobilenet.pb	TensorFlow	0.5484	0.61361	10
yolov3voc.weights	Darknet	0.8926	0.9009	15
yolov3voc_pre_trained.weights	Darknet	0.4577	0.5843	34
yolov4.weights	Darknet	0.9056	0.9670	13
yolov4_pre_trained.weights	Darknet	0.4799	0.5879	24

4.4. Experimental Validation in Good Lightning Conditions. For the final TrafficSensor application, YOLOv3 and YOLOv4 were the selected networks, as they obtained the best results. To validate these final networks, the quality of the whole system has been measured with DetectionMetrics and the created testing dataset. For the sake of comparison, the quality of the initial base-line system, named TrafficMonitor [13] and without deep learning layers, has also been evaluated with the same dataset and measuring tool. In addition, TrafficSensor has also been compared to Deep SORT (Simple, Online and Realtime Tracking with a Deep Association Metric) [67], which is an algorithm commonly used in object tracking. It is an extension to SORT (Simple, Online and Realtime Tracker) [68] that incorporates appearance information through a pretrained association metric. All systems were evaluated with the same good condition videos and images.

The results obtained are given in Table 7. YOLOv4 and YOLOv3 have similar results although YOLOv4 is slightly better. This result was expected as the authors of YOLOv4 [62] indicated that the quality of the detections was superior to that of YOLOv3.

The results of TrafficSensor outperform those of TrafficMonitor. In the successive tests with TrafficMonitor, we have appreciated that it does not work well with distant vehicles (in many cases cars are classified how motorcycles), and it has difficulty to differentiate between car and van. The small vans are confused with cars. This is because the classification is done using 3D models; for this reason, a small 3D van model can be closer to the 3D model of a car than to that of a large van.

In Deep SORT, the YOLOv3 Darknet network trained with our dataset has been used, and the results obtained by TrafficSensor and Deep SORT are very similar. TrafficSensor performs slightly better because it predicts the position of vehicles when they are not detected. Deep SORT uses the Kalman filter to predict and track, but predictions are used to improve detections, not to predict if there is no detection.

Table 7: Results of good conditions video.

System	mAP	mAR
TrafficSensor YOLOv3	0.8926	0.9009
TrafficSensor YOLOv4	0.9056	0.9670
TrafficMonitor	0.4374	0.5940
Deep SORT	0.8164	0.8689

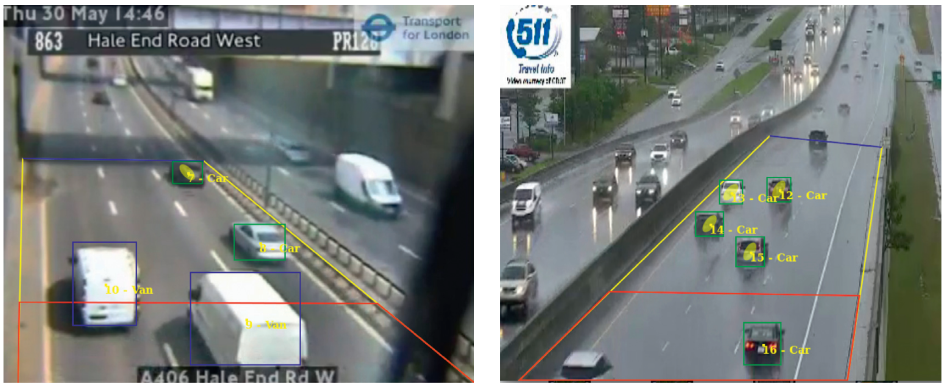
4.5. Experimental Validation in Poor Conditions. The final TrafficSensor system was also evaluated with bad weather conditions and poor quality videos, as shown in Figure 19. Tables 8 and 9 provide the obtained experimental results.

Despite being in rainy conditions, the system is able to work successfully and with very good results. In this test, it can be seen that TrafficMonitor is not so robust because it is not able to function correctly with rain. In the case of Deep SORT, again the results are similar to TrafficSensor.

With all the experimental results gathered, it can be said that TrafficSensor is robust against poor quality images and bad weather conditions. In addition, it is able to continue tracking vehicles when they are far away from the camera. Obviously, it works better with nearby vehicles, as there they are easier to detect, but it is still able to detect and track the distant ones with great quality.

Comparing the experimental results in the videos, the performance with poor quality videos and unfavorable weather conditions is slightly better than for good quality videos. This can be explained since the minimum requirements we set for good quality images are higher than those for bad weather conditions and poor-quality videos. We do not expect the system to be able to detect distant vehicles in bad weather conditions and poor quality videos. It is not even easy for humans to classify such vehicles. The images in the dataset have been labelled following this approach.

When evaluating the results obtained by Deep SORT, they are similar to those of TrafficSensor. TrafficSensor has



(a) (b)

Figure 19: TrafficSensor with poor resolution (a) and bad weather (b) videos.

Table 8: Results of bad weather video.

System	mAP	mAR
TrafficSensor YOLOv3	0.9899	0.9926
TrafficSensor YOLOv4	0.9904	0.9949
TrafficMonitor	0.2407	0.3162
Deep SORT	0.9801	0.9824

Table 9: Results of poor quality video.

System	mAP	mAR
TrafficSensor YOLOv3	0.9439	0.9444
TrafficSensor YOLOv4	0.9902	0.9911
TrafficMonitor	0.4479	0.6303
Deep SORT	0.8852	0.8910

Table 10: Processing time.

Function	With YOLOv3 (ms/call)	With YOLOv4 (ms/call)
Image processing	10	10
Detection algorithm	15	13
Tracking algorithm	18	18

greater precision since in cases where the neural network is not capable of detecting, it predicts such detection using the tracking algorithm.

4.6. Processing times. In the TrafficSensor system, three main processes can be identified: image processing (obtaining images, displaying images, and obtaining data from the delimited road), detection, and tracking. Their computing time performance, both with YOLOv3 and YOLOv4, has been monitored and evaluated. Table 10 provides the obtained results.

5. Conclusion

TrafficSensor system is a solution for vehicle surveillance using deep learning. It is based on a previous nondeep learning solution, named TrafficMonitor [13]. The old solution was based on volumetric 3D patterns, SVM for vehicle classification and background subtraction. This system was able to distinguish between 5 possible classes (motorcycles, cars, vans, buses, and trucks). All these steps were replaced by a neural network for detection and classification. Four state-of-the-art network models have been experimentally tested, even coming from different neural frameworks (Keras, TensorFlow, and Darknet) and with different types of images. The proposed deep learning system classifies the vehicles based on 7 classes: motorcycles, cars, vans, buses, small trucks, trucks, and tank trucks.

A new dataset was created to train and evaluate the new system, including a variety of images such as poor quality images or adverse weather conditions besides the typical good lightning images. TrafficSensor has proven to be robust to bad weather conditions, blurred or low resolution traffic images. This improvement was achieved, thanks to training with the new extensive dataset and the combination of spatial correspondence tracking and KLT tracking on the deep learning-based detections.

Both the YOLOv3 and YOLOv4 networks have been selected for TrafficSensor for their great results. Although, YOLOv4 obtains better results in terms of quality and speed than YOLOv3.

In addition, a new opensource tool has been created to quantitatively and automatically measure the quality of several neural networks for the visual detection task using large datasets. It supports the most widely used neural frameworks (PyTorch, TensorFlow, Keras, and Darknet) and the most common dataset formats in object detection (YOLO, COCO, ImageNet, and Pascal VOC). It measures some useful detection statistics such as IntersectionOver-Union, precision, recall, and inference times. It is publicly available.

As future lines, we intend to test more new state-of-the-art network models for visual object detection, to extend the custom dataset with more images of bad quality or bad lightning conditions, of incoming traffic flow, and to explore the use of attention-based models. In addition, we plan to use DetectionMetrics tool in the medical images domain.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request. Part of them, the GRAM Road-Traffic Monitoring database, comes from a third party source which has been properly cited [32]. In addition, the source code of the Detection-Metrics tool, which has been used for experiments, is publicly available at https://github.com/JdeRobot/DetectionMetrics.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This research was partially funded by the Community of Madrid in the framework of RoboCity2030-DIH-CM (2019–2022): RoboCity2030—Madrid Robotics Digital Innovation Hub, Programa de Actividades de I+D entre Grupos de Investigación de la Comunidad de Madrid en Tecnologías 2018. Project reference: S2018/NMT-4331.

References

[1] M. Hodlmoser, B. Micusik, M. Liu, M. Pollefeys, and M. Kampel, "Classification and pose estimation of vehicles in videos by 3d modeling within discrete-continuous optimization," in *Proceedings of the 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012 Second International Conference*, pp. 198–205, Zurich, Switzerland, October 2012.

[2] N. C. Mithun, N. U. Rashid, and S. M. Rahman, "Detection and classification of vehicles from video using multiple time-spatial images," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, 2012.

[3] N. Thakoor and B. Bhanu, "Structural signatures for passenger vehicle classification in video," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, 2013.

[4] Y. Yang, Y. Ming, Y. Gang, and Z. Yandong, "Length-based vehicle classification in multi-lane traffic flow," *Transactions of Tianjin University*, vol. 17, pp. 362–368, 2011.

[5] W. Yulong, J. Guo, and Z. Qian, "Real-time vehicle classification based on eigenface," in *Proceedings of the Consumer Electronics, Communications and Networks (CECNet)*, pp. 4292–4295, Xianning, China, April 2011.

[6] T. Zhigang and C. Taylor, "A multimodal temporal panorama approach for moving vehicle detection, reconstruction and classification," *Computer Vision and Image Understanding*, vol. 117, no. 12, pp. 1724–1735, 2013.

[7] N. Buch, S. A. Velastin, and J. Orwell, "A review of computer vision techniques for the analysis of urban traffic," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 3, pp. 920–939, 2011.

[8] N. K. Kanhere and S. T. Birchfield, "A taxonomy and analysis of camera calibration methods for traffic monitoring applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 2, pp. 441–452, 2010.

[9] S.-H. Yu, J.-W. Hsieh, Y.-S. Chen, and W.-F. Hu, "An automatic traffic surveillance system for vehicle tracking and classification," *Image Analysis*, vol. 2749, pp. 379–386, 2003.

[10] D. Koller, J. Weber, and J. Malik, "Robust multiple car tracking with occlusion reasoning," in *Proceedings of the Computer Vision-ECCV'94*, pp. 189–196, Stockholm, Sweden, June 1994.

[11] C. C. C. Pang, W. W. L. Lam, and N. H. C. Yung, "A novel method for resolving vehicle occlusion in a monocular traffic-image sequence," *IEEE Transactions on Intelligent Transportation Systems*, vol. 5, no. 3, pp. 129–141, 2004.

[12] C. C. C. Pang, W. W. L. Lam, and N. H. C. Yung, "A method for vehicle count in the presence of multiple-vehicle occlusions in traffic images," *IEEE Transactions on Intelligent Transportation Systems*, vol. 8, no. 3, pp. 441–459, 2007.

[13] R. Kachach and J. M. Cañas, "Hybrid 3d and SVM approach for automatic vehicle tracking and classification using a single camera," *Journal of Electronic Imaging*, vol. 25, 2016.

[14] Y. Benjamin, Y. Wang, and S. Zhu, "Reconfigurable templates for robust vehicle detection and classification," in *Proceedings of the Applications of Computer Vision (WACV)*, pp. 321–328, Breckenridge, CO, USA, January 2012.

[15] Z. Chen and T. Ellis, "Multi-shape descriptor vehicle classification for urban traffic," in *Proceedings of the International Conference on Digital Image Computing: Techniques and Applications*, pp. 465–461, Noosa, QLD, Australia, December 2011.

[16] Z. Chen, T. Ellis, and S. Velastin, "Vehicle type categorization: a comparison of classification schemes," in *Proceedings of the 14th International IEEE Conference on Intelligent Transportation Systems*, pp. 74–79, Washington, DC, USA, October 2011.

[17] K. Yousaf, A. Iftikhar, and A. Javed, "Comparative analysis of automatic vehicle classification techniques: a survey," *International Journal of Image, Graphics and Signal Processing*, vol. 4, no. 9, pp. 52–59, 2007.

[18] Z. Zhaoxiang Zhang, T. Tieniu Tan, K. Kaiqi Huang, and Y. Yunhong Wang, "Three-dimensional deformable-model-based localization and recognition of road vehicles," *IEEE Transactions on Image Processing*, vol. 21, no. 1, pp. 1–13, 2012.

[19] J. M. Blosseville, C. Krafft, F. Lenior, V. Motyka, and S. Beucher, "New traffic measurement by image processing," in *Proceedings of the IFAC Control, Computers, Communications in Transportation*, pp. 35–42, Paris, France, September 1989.

[20] J. Lai, S. Huang, and C. Tseng, "Image-based vehicle tracking and classification on the highway," in *Proceedings of the Green Circuits and Systems (ICGCS), 2010 International Conference on*, pp. 666–670, Shanghai, China, June 2010.

[21] T. Rodríguez and N. García, "An adaptive, real-time, traffic monitoring system," *Machine Vision and Applications*, vol. 21, no. 4, pp. 555–576, 2009.

[22] L. Chen, J. Hsieh, Y. Yan, and D. Chen, *Vehicle Make and Model Recognition Using Sparse Representation and Symmetrical Surfs*, Elsevier, Amsterdam, Netherlands, 2015.

[23] T. Chalidabhongse, K. Kim, D. Harwood, and L. Davis, "A perturbation method for evaluating background subtraction algorithms," in *Proceedings of the IEEE Joint International Workshop VS-PETS*, pp. 1–7, Nice, France, January 2003.

[24] K. Kim, T. Chalidabhongse, D. Harwood, and L. Davis, "Background modeling and subtraction by codebook construction," in *Proceedings of the ICIP*, pp. 3061–3064, November 2004.

[25] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis, "Real-time foreground-background segmentation using codebook model," *Real-Time Imaging*, vol. 11, no. 3, pp. 172–185, 2005.

[26] L. Unzueta, M. Nieto, A. Cortés, J. Barandiaran, O. Otaegui, and P. Sánchez, "Adaptive multicue background subtraction for robust vehicle counting and classification," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 2, pp. 527–540, 2012.

[27] Z. Zivkovic and F. Van Der Heijden, "Efficient adaptive density estimation per image pixel for the task of background subtraction," *Pattern Recognition Letters*, vol. 27, no. 7, pp. 773–780, 2006.

[28] L. Huang, "Real-time multi-vehicle detection and sub-feature based tracking for traffic surveillance systems," in *Proceedings of the 2010 2nd International Asia Conference on Informatics in Control*, March 2010.

[29] K. Robert, "Night-time traffic surveillance: a robust framework for multi-vehicle detection, classification and tracking," in *Proceedings of the Advanced Video and Signal Based Surveillance*, pp. 1–6, Genova, Italy, September 2009.

[30] Z. Zivkovic, "Improved adaptive Gaussian mixture model for background subtraction," in *Proceedings of the 2004 ICPR 2004 17th International Conference on Pattern Recognition*, pp. 28–31, Cambridge, UK, August 2004.

[31] P. H. Samhitha, A. N. Jyothi, R. Vesapogu, M. Mannem, and S. S. Harsha, "Vehicle detection, tracking and speed measurement for traffic regulation," *International Research Journal of Engineering and Technology (IRJET)*, vol. 04, 2017.

[32] R. Guerrero-Gomez-Olmedo, R. J. Lopez-Sastre, S. Maldonado-Bascon, and A. Fernandez-Caballero, "Vehicle tracking by simultaneous detection and viewpoint estimation," *Natural and Artificial Computation in Engineering and Medical Applications, Lecture Notes in Computer Science*, vol. 7931, pp. 306–316, 2013.

[33] M. Hodlmoser, B. Micusik, M. Pollefeys, M. Liu, and M. Kampel, "Model-based vehicle pose estimation and tracking in videos using random forests," in *Proceedings of the International Conference on 3D Vision*, pp. 430–437, Seattle, WA, USA, June 2013.

[34] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[35] K. Mu, F. Hui, and X. Zhao, "Multiple vehicle detection and tracking in highway traffic surveillance video based on sift feature matching," *Journal of Information Processing Systems*, vol. 12, 2016.

[36] M. Nieto, L. Unzueta, J. Barandiaran, A. Cortés, O. Otaegui, and P. Sánchez, "Vehicle tracking and classification in challenging scenarios via slice sampling," *EURASIP Journal on Applied Signal Processing*, vol. 2011, no. 1, 2011.

[37] Z. Chen, T. Ellis, and S. A. Velastin, "Vehicle detection, tracking and classification in urban traffic," in *Proceedings of the 15th International IEEE Conference on Intelligent Transportation Systems*, Anchorage, AK, USA, September 2012.

[38] Z. Zhu and X. Lu, "An accurate shadow removal method for vehicle tracking," in *Proceedings of the Artificial Intelligence and Computational Intelligence (AICI), 2010 International Conference on*, pp. 59–62, Sanya, China, October 2010.

[39] L. Wang, F. Chen, and H. Yin, *Detecting and Tracking Vehicles in Traffic by Unmanned Aerial Vehicles*, Department of Civil Engineering and Engineering Mechanics, Columbia University, New York, NY, USA, 2016.

[40] L. Huang and M. Barth, "Real-time multi-vehicle tracking based on feature detection and color probability model," in *Proceedings of the Intelligent Vehicles Symposium (IV)*, pp. 981–986, La Jolla, CA, USA, June 2010.

[41] B. Johansson, J. Wiklund, P. Forssén, and G. Granlund, "Combining shadow detection and simulation for estimation of vehicle size and position," *Pattern Recognition Letters*, vol. 30, 2009.

[42] M. J. Leotta and J. L. Mundy, "Vehicle surveillance with a generic, adaptive, 3d vehicle model," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 7, pp. 1457–1469, 2011.

[43] K. D. Baker and G. D. Sullivan, "Performance assessment of model-based tracking," in *Proceedings of the IEEE Workshop on Applications of Computer Vision*, pp. 28–35, Palm Springs, CA, USA, November 1992.

[44] G. Welch and G. Bishop, "An introduction to the Kalman filter," Technical Report TR 95-041, University of North Carolina at Chapel Hill, Chapel Hill, NC, USA, 2006.

[45] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman, "Multiple kernels for object detection," in *Proceedings of the IEEE 12th International Conference on Computer Vision (ICCV)*, Kyoto, Japan, September 2009.

[46] B. Scholkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and beyond*, MIT Press, Cambridge, MA, USA, 2001.

[47] C. Papageorgiou and T. Poggio, "A trainable system for object detection," *International Journal of Computer Vision*, vol. 38, no. 1, pp. 15–33, 2000.

[48] C. P. Papageorgiou, M. Oren, and T. Poggio, "A general framework for object detection," in *Proceedings of the Sixth International Conference on Computer Vision, ICCV'98*, Bombay, India, January 1998.

[49] C. M. Bautista, C. A. Dy, M. I. Mañalac, R. A. Orbe, and M. Cordel, *Convolutional Neural Network for Vehicle Detection in Low Resolution Traffic Videos*, Center for Automation Research College of Computer Studies, De La Salle University, Manila, Philippines, 2016.

[50] Y. Jia, E. Shelhamer, J. Donahue et al., "Caffe: convolutional architecture for fast feature embedding," URL http://caffe.berkeleyvision.org/tutorial/solver.html, 2014.

[51] J. K. Sensa, G. S. Syahra, C. K. Dewa, and Afiahayati, "Traffic congestion detection: learning from CCTV monitoring images using convolutional neural network," in *Proceedings of the INNS Conference on Big Data and Deep Learning*, pp. 291–297, Bali, Indonesia, January 2018.

[52] W. Yang, Z. Li, C. Wang, and J. Li, "A multi-task faster R-CNN method for 3d vehicle detection based on a single image," *Applied Soft Computing Journal*, vol. 95, 2020.

[53] J. Luo, H. Fang, F. Shao, Y. Zhong, and X. Hua, "Multi-scale traffic vehicle detection based on faster RE-CNN with NAS optimization and feature enrichment," *Defence Technology*, vol. 17, 2020.

[54] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: unified, real-time object detection," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 779–788, Las Vegas, NV, USA, June 2016.

[55] J. F. Rajotte, M. Sotir, C. Noiseux, L. P. Noel, and T. Bertiere, "Object counting on low quality images: a case study of near real-time traffic monitoring," in *Proceedings of the 2018 17th IEEE International Conference on Machine Learning and Applications*, Orlando, FL, USA, December 2018.

[56] C. Kwan, D. Gribben, B. Chou et al., "Real-time and deep learning based vehicle detection and classification using pixel-wise code exposure measurements," *Electronics*, vol. 9, 2020.

[57] P. Mahto, P. Garg, P. Seth, and J. Panda, "Refining yolov4 for vehicle detection," *International Journal of Advanced Research in Engineering and Technology (IJARET)*, vol. 11, 2020.

[58] L. Zhang, H. Wang, X. Wang, S. Chen, H. Wang, and K. Zheng, "Vehicle object detection based on improved retinanet," *Journal of Physics: Conference Series*, vol. 1757, 2021.

[59] C. Szegedy, A. Toshev, and D. Erhan, "Deep neural networks for object detection," in *Advances in Neural Information Processing Systems*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds., vol. 26, Curran Associates, Inc., New York, NY, USA, 2013https://proceedings.neurips.cc/paper/2013/file/f7cade80b7cc92b991cf4d2806d6bd78-Paper.pdf URL.

[60] 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.

[61] J. Redmon and A. Farhadi, "Yolov3: an incremental improvement," URL http://arxiv.org/abs/1804.02767, 2018.

[62] A. Bochkovskiy, C. Y. Wang, and H. Y. Liao, "Yolov4: optimal speed and accuracy of object detection," 2020, https://arxiv.org/abs/2004.10934.

[63] C. Y. Wang, H. Y. M. Liao, I. H. Yeh, Y. H. Wu, P. Y. Chen, and J. W. Hsieh, "CSPNet: a new backbone that can enhance learning capability of CNN," in *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 1571–1580, Seattle, WA, USA, June 2020.

[64] 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.

[65] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8759–8768, Salt Lake City, UT, USA, June 2018.

[66] J. Y. Bouguet, *Pyramidal Implementation of the Lucas Kanade Feature Tracker*, Intel Corporation, Microprocessor Research Labs, California, USA, 2000.

[67] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *Proceedings of the 2017 IEEE International Conference on Image Processing (ICIP)*, pp. 3645–3649, Beijing, China, September 2017.

[68] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *Proceedings of the 2016 IEEE International Conference on Image Processing (ICIP)*, Phoenix, AZ, USA, September 2016.

Hindawi
Computational Intelligence and Neuroscience
Volume 2021, Article ID 6789956, 13 pages
https://doi.org/10.1155/2021/6789956

Research Article

Integrated Multiscale Appearance Features and Motion Information Prediction Network for Anomaly Detection

Ting Liu,[1] Chengqing Zhang,[1,2] and Liming Wang[1]

[1]*State Key Lab for Electronic Testing Technology, North University of China, Taiyuan 030051, China*
[2]*College of Mechatronics Engineering, North University of China, Taiyuan 030051, China*

Correspondence should be addressed to Liming Wang; wlm@nuc.edu.cn

Received 2 July 2021; Accepted 6 October 2021; Published 20 October 2021

Academic Editor: Jose Carlos Rangel

The rise of video-prediction algorithms has largely promoted the development of anomaly detection in video surveillance for smart cities and public security. However, most current methods relied on single-scale information to extract appearance (spatial) features and lacked motion (temporal) continuity between video frames. This can cause a loss of partial spatiotemporal information that has great potential to predict future frames, affecting the accuracy of abnormality detection. Thus, we propose a novel prediction network to improve the performance of anomaly detection. Due to the objects of various scales in each video, we use different receptive fields to extract detailed appearance features by the hybrid dilated convolution (HDC) module. Meanwhile, the deeper bidirectional convolutional long short-term memory (DB-ConvLSTM) module can remember the motion information between consecutive frames. Furthermore, we use RGB difference loss to replace optical flow loss as temporal constraint, which greatly reduces the time for optical flow extraction. Compared with the state-of-the-art methods in the anomaly-detection task, experiments prove that our method can more accurately detect abnormalities in various video surveillance scenes.

1. Introduction

Due to the corona virus disease of 2019 (COVID-19) outbreak, many countries have been accelerating the construction of smart cities and public-safety systems [1] to efficiently manage surrounding circumstances. As part of these systems, traditional video surveillance systems rely on manual monitoring to find abnormalities in massive video data. This operation increases working time, labor costs, and misjudgments. Therefore, automatic detection of anomalous behaviors [2] has attracted increasing researcher attention because of its potential application values. An intelligent video surveillance system aims to provide a supervisor with precise anomaly cues to deal with abnormal events as soon as possible. However, it is a highly challenging task in the computer vision field, because anomaly detection suffers two core issues. First, only normal samples are readily available during the training phase because of the rare occurrence of abnormal events in most cases. Second, anomalous events are various and complicated, and the definition of "abnormality" heavily depends on the context; hence, there is no standard definition. It is difficult to mark abnormal events and detect these behaviors using supervised technology.

To solve the aforementioned problems, most state-of-the-art approaches adopt unsupervised techniques and then use regular events as training samples to train the model. When the test sample deviates significantly from the learned model, it is detected as an anomaly. To date, the large variety of anomaly-detection methods can be roughly divided into two types: (1) hand-crafted feature approaches and (2) deep-learning approaches. In hand-crafted feature methods, the core idea is mainly to adopt hand-crafted features to represent video sequences. These features include trajectory features [3] and low-level features (e.g., histograms of oriented gradients [4], histograms of optical flow [5], and 3D gradients [6]). They are heavily dependent on the feature-extraction process and expert knowledge, which directly limit the accurate representation of complex feature patterns and affect the accuracy of anomaly detection. Deep-learning approaches commonly use reconstruction error-based methods. These methods follow the rule that normal events produce a small reconstruction error, whereas abnormal events generate a large error. They evaluate the anomaly based on the consistency between the generated and the input frames. Specifically, Hasan et al. [7] presented an approach based on the auto-encoder that reconstructs regularities with low error but incurs higher reconstruction error for irregularities. However, because a convolution operation is only used for feature extraction, this structure cannot model temporal information in a long video sequence. Consequently, Chong and Tay [8] and Luo et al. [9] added convolutional long short-term memory (ConvLSTM) layers to the auto-encoder for performing the memory of temporal information. Li Chang [10] presented a multivariate Gaussian fully convolution adversarial auto-encoder (MGFC-AAE) to model gradient and optical flow patches for anomaly detection. George et al. [11] proposed a non-uniform spatiotemporal region resembling parallelepipeds to extract the histogram of optical flow orientation and magnitude features. These approaches simultaneously modelled spatial and temporal features from the input data, making them more suitable for video analysis. Nevertheless, it is challenging to obtain a large reconstruction error for anomalies owing to the powerful learning capacity of a deep neural network. Moreover, because of the self-reconstructed generated frames, the methods identify anomalies regardless of context information. Therefore, high missed and false detection phenomena occur while executing these methods.

Considering the shortcomings of reconstruction approaches, some researchers have begun to use video-prediction algorithms, namely, future-frame prediction based on a sequence of previous video frames, to detect abnormal behaviors. These methods agree with the idea that normal events are predictable, whereas abnormal events are unpredictable. By only training regular events to obtain a prediction model, anomalies in videos refer to events that rarely or should not occur in a particular scenario. For example, Munawar et al. [12] created a deep prediction network to detect the abnormal operation behaviors of industrial robots. Villegas et al. [13] combined LSTM and analogy-based encoder-decoder networks to tackle long-term video-prediction tasks from a hierarchical perspective. Additionally, Zhao et al. [14] proposed a spatiotemporal auto-encoder involving the three-dimensional (3D) convolution for video anomaly detection. Nevertheless, these methods based on an auto-encoder structure use only single-scale information from the previous layer in the decoding process, leading to the detailed information loss for the different-size objects in the videos. Thus, Liu et al. [15] proposed a method to predict future frames on the basis of U-Net, which can effectively retain the multiscale structural characteristics of the input frames by the skip connection. However, the conventional U-Net cannot adequately consider the motion continuity between video frames.

Motivated by the aforementioned anomaly detection task, it is necessary to sufficiently consider multiscale spatial features and temporal continuity for recognizing abnormal behaviors. Recently, lots of works have achieved great detection performance by using multiscale features of images; for example, Gao et al. [16] adopted multiscale single-stage object detector for pose detection in the classroom scene. Oh et al. [17] proposed multiscale convolutional recurrent neural network for inspecting and classifying bearing fault defects. The literatures [18, 19] used multiview receptive field network for foreground detection. Owing to the camera position and angle, objects multiscale features extraction can effectively improve the performance of target detection. In this paper, we propose a novel spatiotemporal prediction network, i.e., STP-net, which fuses the multiscale appearance features and motion information extraction module. The main idea is to utilize the network to model the video content and internal dynamic changes by training the ordinary events accurately. If the test-video prediction frame is significantly different from the actual frame, an abnormality is detected. First, we use the HDC module [20] to extract multiscale spatial features and learn the objects scale variations. Then, we adopt the DB-ConvLSTM [21] module to memorize the temporal information and obtain the complex motions features between consecutive frames. Finally, we perform the predicted future frame from the spatial and temporal dimensions. At the same time, the literatures [22, 23] showed that RGB difference is a valid substitute for optical flow [24] as a new type of temporal loss. This operation could achieve a similar effect but significantly reduce the computational cost to extract optical flow information.

Specifically, the main contributions of our work are as follows:

(1) Starting from the second downsampling of U-Net, the HDC module acts on the previous convolution layer of each downsampling layer to increase the convolution kernel receptive field, making it easy to retain more data detailed information and improve the representational capacity of the model.

(2) At the end of the encoding process of U-Net, the DB-ConvLSTM strategy can take full advantage of the relationship between consecutive frames to extract detailed temporal information, which can strengthen the temporal continuity between the video frames and effectively improves the accuracy of the prediction results.

(3) Experimental results on several public benchmark datasets indicate the superior ability of our method compared with the state-of-the-art approaches in the abnormality detection task.

The remainder of this paper is organized as follows. Section 2 provides the overall framework of the proposed method. Section 3 elucidates and discusses the experimental validation through a series of primary public datasets. Finally, Section 4 summarizes the general conclusions and discusses future research directions.

2. Proposed Method

As shown in Figure 1, the overall framework of our method can be divided into two parts: video prediction and anomaly

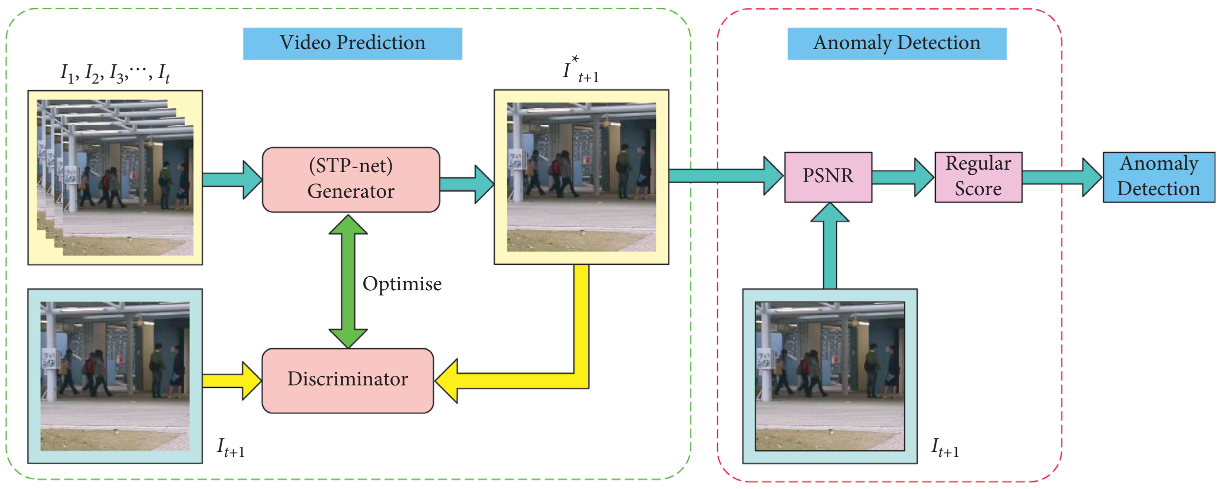


FIGURE 1: Overall framework of the proposed method.

detection. The first part aims to train a generator network to predict future frames. To generate a high-quality prediction frame, we use the generative adversarial network (GAN) [25] and several loss functions to optimize our network model. We treat the STP-net as generator network (G) and then adopt frames $(I_1, I_2, I_3, \ldots, I_t)$ before the current frame I_{t+1} as the input tensor, and the predicted frame I^*_{t+1} as the output tensor. For the discriminative network (D), we choose PatchGAN [26] to strengthen the recognizing ability between the actual and generated frames. Finally, we use the total objective optimization function to minimize the distance between the predicted frame and the target frame, making I^*_{t+1} closer to I_{t+1}. In the second part, we employ the pretrained model to judge the extent of abnormality by calculating each frame's regular score. Next, we will illustrate the different components of the proposed framework in detail.

2.1. Video Prediction. On the basis of U-Net structure, the details of STP-net are presented in Figure 2. We add HDC module to extract multiscale spatial features of the training samples and then insert DB-ConvLSTM to handle temporal information between the continuous T frames in a nonlinear manner. The network comprises an encoding path and a decoding path. The input and output size of the network are both $256 \times 256 \times 3$. The kernel sizes of all convolution and deconvolution are set to 3×3 and the maxpool layers are set to 2×2.

2.1.1. Multiscale Features Extracted Strategy. The objects forms and sizes are different owing to the camera position and angle. Inspired by the HDC applied in the semantic segmentation field, it is essential to consider multiscale feature information. Meanwhile, the multiple downsampling operations of the U-Net will lead to the severe loss of spatial detailed information. In order to improve the network's learning ability, we should not only consider extracting multiscale spatial information, but also consider compensating for the loss due to the downsampling operation; thus, starting from the second downsampling, the HDC module acts on the previous convolution layer of each downsampling layer to retain more image detail information. The reason why HDC is not used before the first downsampling layer is that several convolution operations before first downsampling will not cause a lot of loss to image information.

The structure of HDC module is shown in Figure 3. The input feature maps are fed into three different branches. These branches are used to acquire the different size of receptive field and automatically extract multiscale features through a set of dilated convolutions with different dilation rates. It is also worth mentioning that a small dilation rate is fit for extracting features of small objects, while a large dilation rate is fit for obtaining features of large objects. Finally, the features from each branch are concatenated with the input feature maps for enhancing contextual information and multiscale spatial features representation.

2.1.2. Temporal Information Extracted Strategy. The current anomaly detection methods usually adopt three-dimensional (3D) convolution or ConvLSTM [27] to extract the temporal correlation of the input data. The 3D convolution requires more computational time to process a large number of model parameters. Therefore, lots of researchers choose ConvLSTM structure for time modelling. However, the ConvLSTM can only remember the sequence data in the forward direction. According to study [21, 28], it is evident that considering both forward and backward feature information is important and complementary for predicting future frames. Thus, we use DB-ConvLSTM module to capture more comprehensive spatiotemporal characteristics.

The input mode of our network is different from existing methods that conventionally stack T consecutive frames together into a network. In these methods, all the T frames are connected to each channel in the first output feature map, which results in the collapse of temporal information [29]; thus, we input T frames into the encoder network one by one to generate corresponding feature maps. As shown in Figure 4, the DB-ConvLSTM structure includes a

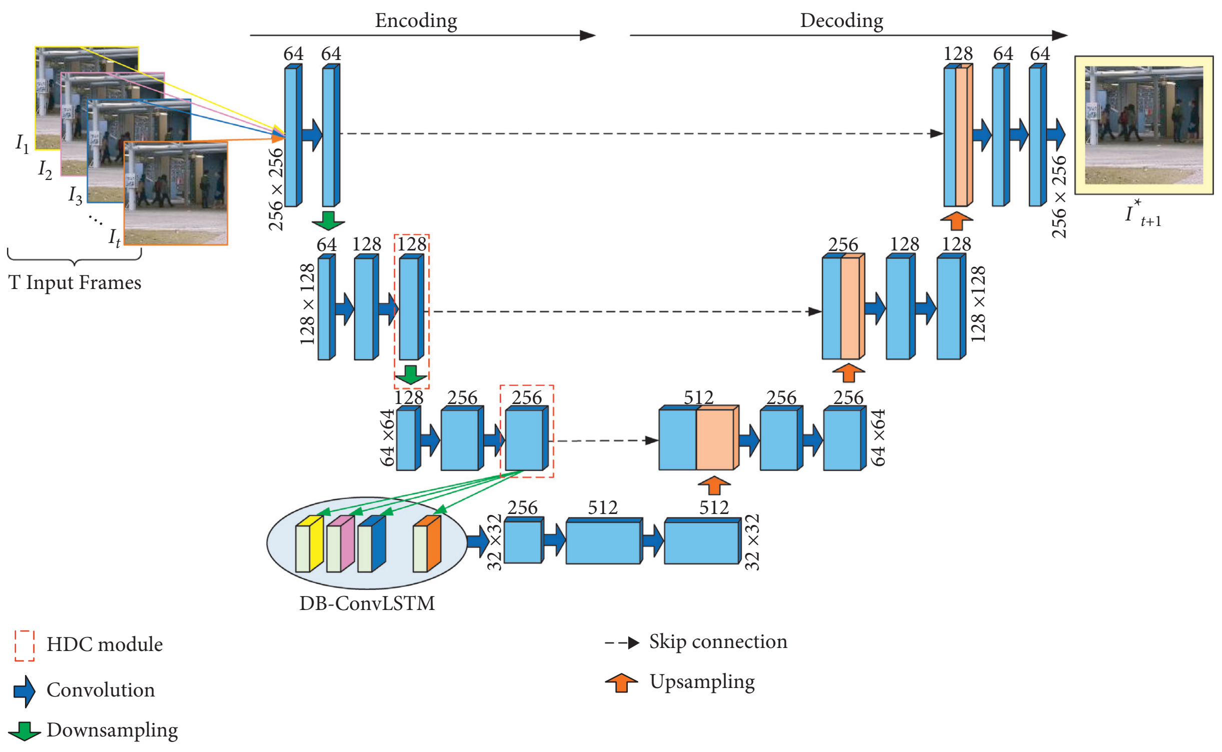


Figure 2: The structure of STP-net. The resolutions of feature maps are equal in the same layer.

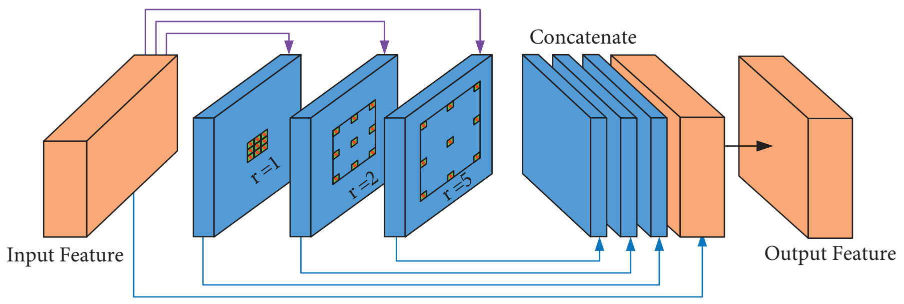


Figure 3: The structure of HDC module. The sizes of input feature maps are $128 \times 128 \times 128$ and $64 \times 64 \times 256$. The dilation rates are set to 1, 2, and 5, respectively.

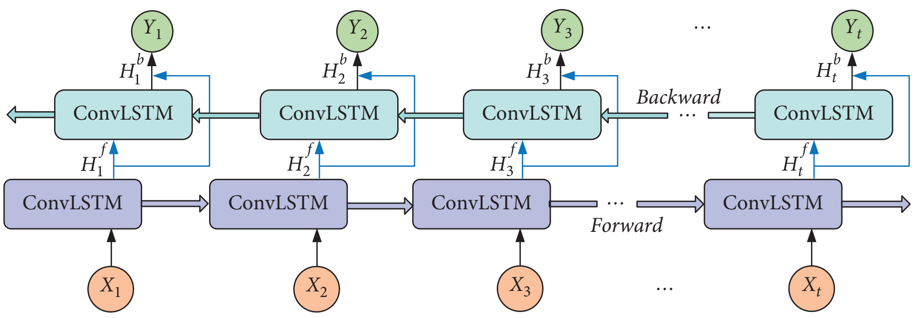


Figure 4: The structure of DB-ConvLSTM module.

shallow forward layer and a deeper backward layer. Specifically, $\{H_t^f\}$ denotes the corresponding outputs of forward sequential feature maps from the ConvLSTM units in the forward layer. The deeper backward layer receives the forward sequential outputs $\{H_t^f\}$ to generate $\{H_t^b\}$ corresponding outputs of backward sequential feature maps. Then, we use equation (1) to process the forward and the backward features maps to obtain the final output sequence $\{Y_t\}$. Finally, the information can exchange between the forward and backward directional ConvLSTM units to capture more powerful and complementary spatiotemporal features. As shown in Figure 4, we feed the last output Y_t containing both spatial features and relevant temporal features into the decoding process.

$$Y_t = \tanh\left(W_y^{Hf} * H_t^f + W_y^{Hb} * H_t^b + b\right). \tag{1}$$

2.1.3. Loss Function. We used spatial and temporal constraints to optimize the model and minimize the difference between the predicted frame and its ground truth. The intensity constraint can guarantee the similarity of all pixels in the RGB space, and the gradient constraint can sharpen the generated images. Therefore, we chose intensity and gradient constraints as spatial constraint to promote the predicted frames I^* to be consistent with the corresponding ground truth I. Moreover, the temporal loss defined as the RGB difference between the prediction frame and the ground truth guarantees the correctness of motion prediction for anomaly detection. The intensity loss, gradient loss, and temporal loss are defined as equations (2)–(4), respectively.

$$L_{\text{int}}(I^*, I) = \|I^* - I\|_2^2, \tag{2}$$

$$L_{gd}(I^*, I) = \sum_{i,j} \left\| \left|I_{i,j}^* - I_{i-1,j}^*\right| - \left|I_{i,j} - I_{i-1,j}\right| \right\|_1 + \left\| \left|I_{i,j}^* - I_{i,j-1}^*\right| - \left|I_{i,j} - I_{i,j-1}\right| \right\|_1, \tag{3}$$

$$L_{rgb}(I^*, I) = \left\| \left|I_{t+1}^* - I_t\right| - \left|I_{t+1} - I_t\right| \right\|_1. \tag{4}$$

We also leveraged GAN to constrain the training process owing to its excellent image generation [30] and video-prediction [31] performance in recent years. Specifically, G attempts to generate future frames that are as realistic as possible, whereas D aims to distinguish the frames generated by G. Ideally, the goal of the GAN is to reach the Nash equilibrium. When training D, the procedure aims to classify $I*$ into class 0 and I into class 1, where 0 represents the generated frame, and 1 indicates the genuine frame. The loss function used to train D is imposed as equation (5). When training G, the goal is to let the generated frames $I*$ classified into class 1 by D. Then, the adversarial loss for G is defined as shown in equation (6):

$$L_{adv}^D(I^*, I) = \frac{1}{2}(D(I^*) - 0)^2 + \frac{1}{2}(D(I) - 1)^2, \tag{5}$$

$$L_{adv}^G(I^*) = \frac{1}{2}(D(I^*) - 1)^2. \tag{6}$$

To obtain a well-trained model that has a better ability to identify abnormalities, we considered all the aforementioned constraints, such as spatial, temporal, and adversarial training loss, into our final objective function (7). During training D, we fixed the weights of G to optimize objective function (8).

$$L_G = \alpha_{\text{int}} L_{\text{int}} + \alpha_{gd} L_{gd} + \alpha_{rgb} L_{rgb} + \alpha_{adv} L_{adv}^G, \tag{7}$$

$$L_D = L_{adv}^D, \tag{8}$$

where α_{int}, α_{gd}, α_{rgb}, and α_{adv} are coefficients for the corresponding constraints, respectively.

2.2. Anomaly Detection. After training the model to represent regular events in video sequences, we used the difference between the predicted frame I^* and ground truth I for anomaly prediction. To the best of our knowledge, Peak Signal to Noise Ratio (PSNR) [32] is widely used to assess the image quality as follows:

$$\text{PSNR}(I^*, I) = 10 \log_{10} \frac{[\max I^*]^2}{1/N \sum_{i=0}^{N} (I_i^* - I_i)^2}, \tag{9}$$

where I^* represents the predicted frame, I denotes the corresponding ground truth, $\max_{I_*}$ represents the maximum value of the image intensities, N represents the total number of pixels, and i represents the pixel index.

In the test phase, we chose the PSNR to evaluate the predicted frame. A higher PSNR value means that the predicted frame is more similar to its ground truth and indicates that it is more likely to be a regular event and vice versa. For comparison, we normalized the PSNR of all frames in each test video to the range [0, 1], and the regular score can be calculated as

$$S(t) = \frac{\text{PSNR}(I_t^*, I_t) - \min_t \text{PSNR}(I_t^*, I_t)}{\max_t \text{PSNR}(I_t^*, I_t) - \min_t \text{PSNR}(I_t^*, I_t)}, \tag{10}$$

where the $\min_t$PSNR and $\max_t$PSNR are the minimum and maximum values of the PSNR in every test video frame, respectively.

3. Experimental Results and Discussion

In this section, we validate the proposed method performance on publicly available benchmark datasets, including the Chinese University of Hong Kong (CUHK) Avenue dataset [33] and the University of California San Diego (UCSD) Pedestrian dataset [34]. We further utilize the recorded real video data to verify the robustness of our model. The proposed framework was implemented by PyTorch and supported by an NVIDIA Tesla V100.

3.1. Evaluation Metric. To validate the effectiveness of the proposed method, we followed the performance evaluation of frame-level criteria. We selected the receiver operating characteristic (ROC) curve as an indicator to evaluate the anomaly detection algorithms. The ROC curve is obtained by gradually changing the threshold and calculating the true positive rate (TPR) and the false positive rate (FPR). In this study, our approach is compared with the existing anomaly-detection methods using the area under the curve (AUC) and equal error rate (EER). Higher AUC values and lower EER values indicated better anomaly detection performance. The relationship between AUC and EER is illustrated in Figure 5.

3.2. Dataset Description. CUHK Avenue Dataset is collected on Campus Avenue at the Chinese University of Hong Kong and includes 16 training videos (15,328 training frames) and 21 testing videos (15,324 testing frames). Each video-frame resolution is 360×640 pixels, and the frame rate for each video clip is 25 frames per second. Normal events are mainly behaviors of pedestrians walking on the sidewalk. The anomalies include abnormal events, such as running, loitering, and throwing objects.

UCSD Dataset contains two subsets, Ped1 and Ped2, which comprise videos collected by the University of California San Diego from public pedestrian areas taken at different viewing angles. Ped1 comprises 34 training scenes and 36 testing scenes with a frame resolution of 238×158 pixels. Ped2 includes 16 training scenes and 12 testing scenes with a frame resolution of 360×240 pixels. Ped1 and Ped2 have the same definitions of normal and abnormal events. In regular videos, some pedestrians walk on the sidewalk. However, in abnormal cases, these are bicycles, vehicles, skateboarders, and wheelchairs crossing pedestrian areas.

3.3. Training Details. For the training details of our model, we adopted Adam [35] to train the network for parameter optimization. We set T to 4, used a random clip of five sequential frames, and set the mini-batch size to 4. For greyscale datasets, we set the learning rates of the generator and discriminator to 0.0001 and 0.00001, while we set them to 0.0002 and 0.00002 for color-scale datasets. For different datasets, the coefficient factors α_{int}, α_{gd}, α_{rgb}, and α_{adv} were slightly different.

3.4. Performance Analysis of the Proposed Method. We analyze the corresponding experimental results of different datasets. For a better illustration, in Figure 6, specific events are chosen to display the anomaly detection results from the seventh test video on the Avenue dataset. Figure 6(a) shows the corresponding ground truth. Figure 6(b) presents the difference between the ground truth and the corresponding predicted frames. Figure 6(c) displays the relationship between the test video frames and the regular score. The blue blocks represent the ground truth annotation of frames containing abnormal events, and the red line represents the regular score of every frame. As shown in Figure 6(c), higher regular scores represent the usual events. In comparison, the lower regular scores corresponding to the blue area are the abnormal events shown in Figure 6(a) (e.g., the child running from a different direction). When executing the prediction model, our method has learned prior information and then predicts what will happen next. Under the pedestrian street scene, the model gains the appearance and motion features of walking persons from the training samples. As shown in Figure 6(b), when the testing frames of a running person are fed into the model, it can only predict a person while walking, which generates a big difference (labelled with a red rectangle) between the predicted frame and the ground truth.

The size and shape of the objects may change because of the different position and angle of the camera. More specifically, Figures 7 and 8 show the detection results of anomalous events from different video angles on the UCSD Ped1 and Ped2 datasets. The illustrations of these figures are similar to Figure 6. As shown in Figures 7(a) and 7(b) and 8(a) and 8(b), objects located close to the camera appear to be larger than those far from it, although they are the same objects. Moreover, we can see that our method can easily detect abnormal events (e.g., cars and cyclists) from different situations. As shown in Figures 7(c) and 8(c), the lower regular scores are consistent with the ground truth labelled as abnormal events (e.g., the cars in the Ped1 19th test video and the cyclists in the Ped2 2nd test video). Higher regular scores indicate normal events. After analyzing the experimental data, we find that our method is robust when facing these different types of spatial features, because it uses the advantages of HDC module to pay more attention to the multiscale spatial characteristics.

To validate that our method is actually working on a real scenario, we recorded the street scene next to our building and verified the proposed model. The illustrations of these figures are similar to Figure 6. As shown in Figures 9(a) and 9(b), we can see that our method can easily detect abnormal events (e.g., car) from the recorded real video. As shown in Figure 9(c), higher regular scores represent normal activities. The lower regular scores are consistent with the ground truth labelled as abnormal activity.

Additionally, Figure 10 shows the experimental failure case of detecting anomalies in the initial stage on the UCSD

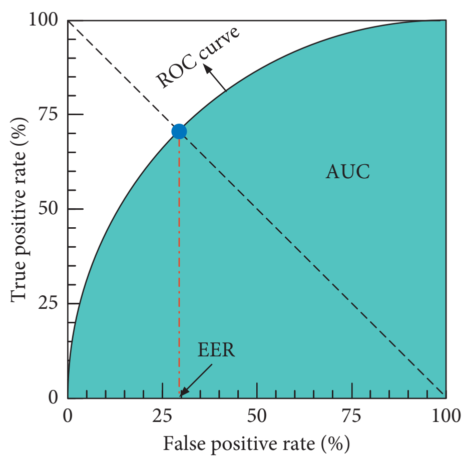


FIGURE 5: Relationship between AUC and EER.

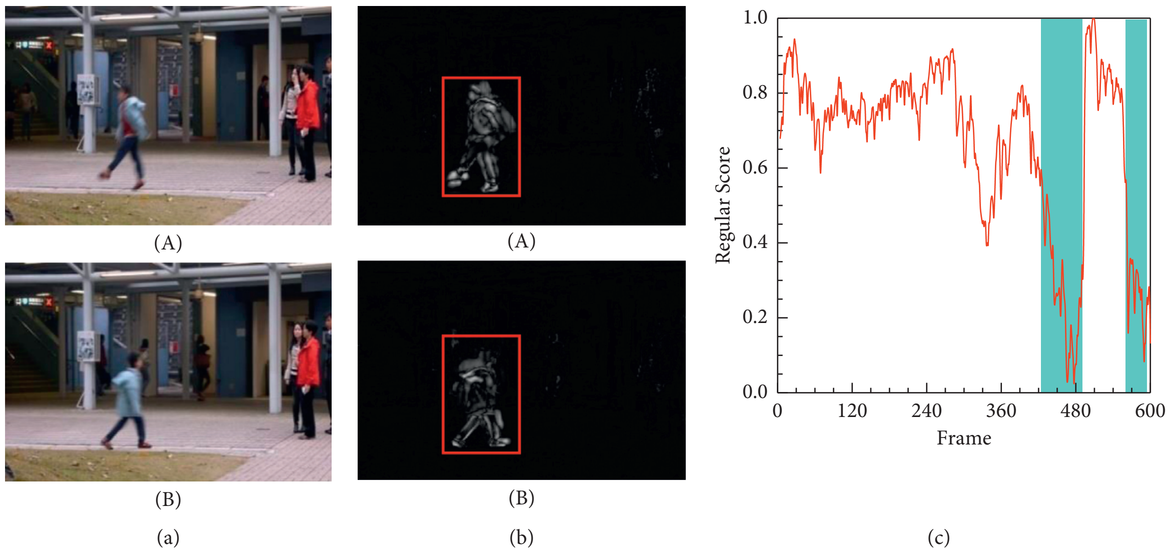


FIGURE 6: Frame-level evaluation results on Avenue 7th test video. (a) Ground truth of the 465th frame (A) and 593rd frame (B) labelled as abnormal. (b) Difference between the ground truth and the corresponding predicted frame. (c) Relationship between the test video frames and the regular score.

Ped2 dataset. As shown in Figure 10(a), we can see that abnormal events (e.g., occluded cyclist) cannot be detected, but the cyclist can be detected without occlusion. The higher regular scores are consistent with the ground truth labelled as abnormal events in the initial phase. As shown in Figure 10(b), the difference (occluded cyclist labelled with a red rectangle) between the ground truth and the corresponding generated frame is ambiguous, but the other one is clear. After analyzing the experimental data, it is worth mentioning that our method might not perform well, because the abnormal events could be temporally occluded by other objects in the video. The main attention of our future work is to solve the problem caused by occlusion, by exploiting visual tracking technology to tackle the miss detection in highly occlusion scenes.

3.5. Performance Comparison of Different Methods. To intuitively display the changing trend of ROC curves of different methods in terms of the frame-level criterion, Figure 11 depicts the results of our method compared with three typical approaches, e.g., MGFC-AAE [10], Baseline [15], and 150FPS [33] on the Avenue dataset. We can observe that the ROC curve of our method is significantly higher than that of the other algorithms. Table 1 presents a quantitative comparison of our method with other recently published approaches for AUC values. Compared with these approaches, the proposed method achieved the highest AUC value, which reached 86.4%, demonstrating good performance.

Figures 12(a) and 12(b) depict the comparison results of the ROC curves of different methods on the UCSD dataset. We chose some deep-learning algorithms [10, 15] and

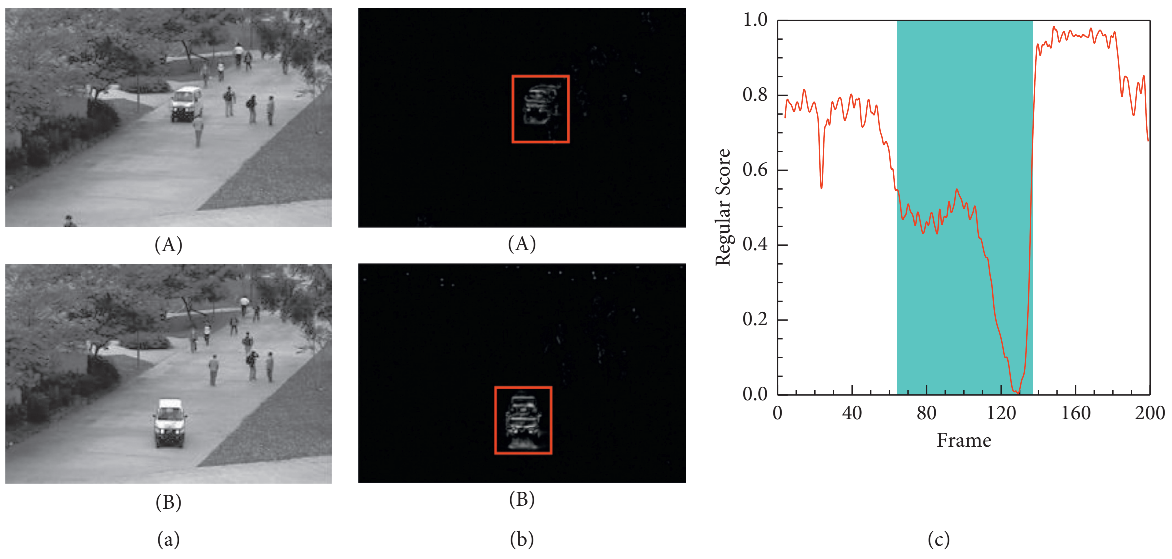


FIGURE 7: Frame-level evaluation results on Ped1 19th test video. (a) Ground truth of the 91st frame (A) and 118th frame (B) labelled as abnormal. (b) Difference between the ground truth and the corresponding predicted frame. (c) Relationship between the test video frames and the regular score.

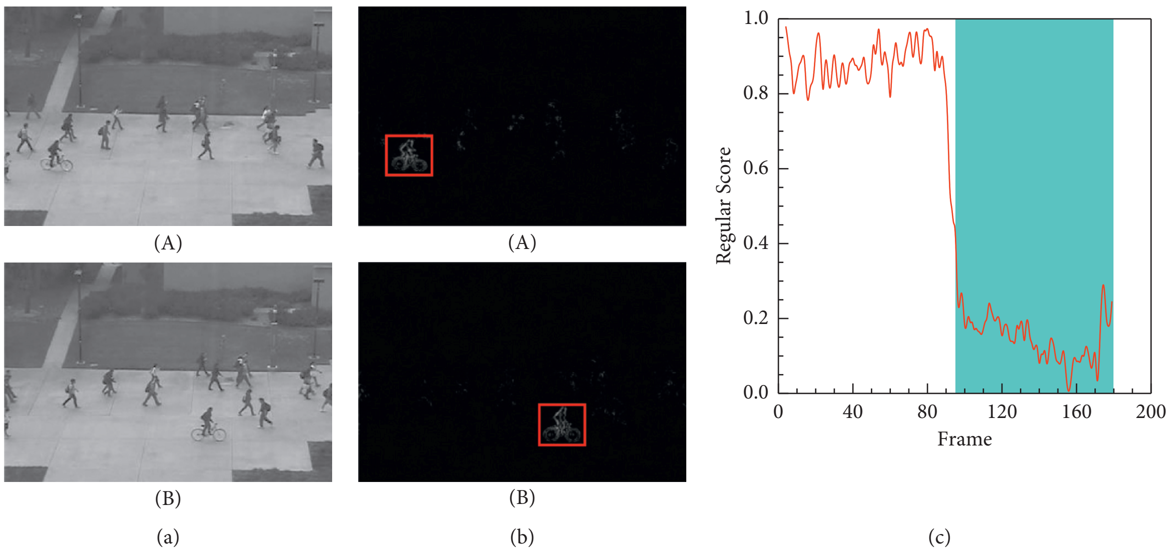


FIGURE 8: Frame-level evaluation results on Ped2 2nd test video. (a) Ground truth of the 109th frame (A) and 160th frame (B) labelled as abnormal. (b) Difference between the ground truth and the corresponding predicted frame. (c) Relationship between the test video frames and the regular score.

traditional methods [34, 38], e.g., MGFC-AAE [10], Baseline [15], mixtures of dynamic textures (MDT) [34], and motion energy model [38]. From the comparison, we can see that our method outperforms most of the existing methods. The experimental results further demonstrate the superiority of the deep-learning methods compared with the traditional methods. Table 2 lists the detailed quantitative comparison data of the different algorithms in the aspect of the AUC metric. We set the literature [15] as the baseline during the evaluation phase because of its excellent performance for anomaly detection based on a prediction network. In detail, our method raises 1.3% and 0.9% for Ped1 and Ped2 datasets compared with Baseline [15]. In conclusion, our method is effective for detecting anomalies on the UCSD dataset.

Through the aforementioned comparison, the proposed method achieved better results in various video surveillance scenes; the AUC value obtained by our model is superior to most existing models. For a more comprehensive analysis, we also adopted EER as the evaluation metric. Table 3 presents the detection results obtained from the proposed method as well as other methods. It can be seen from the data that our method reaches a lower EER compared with all

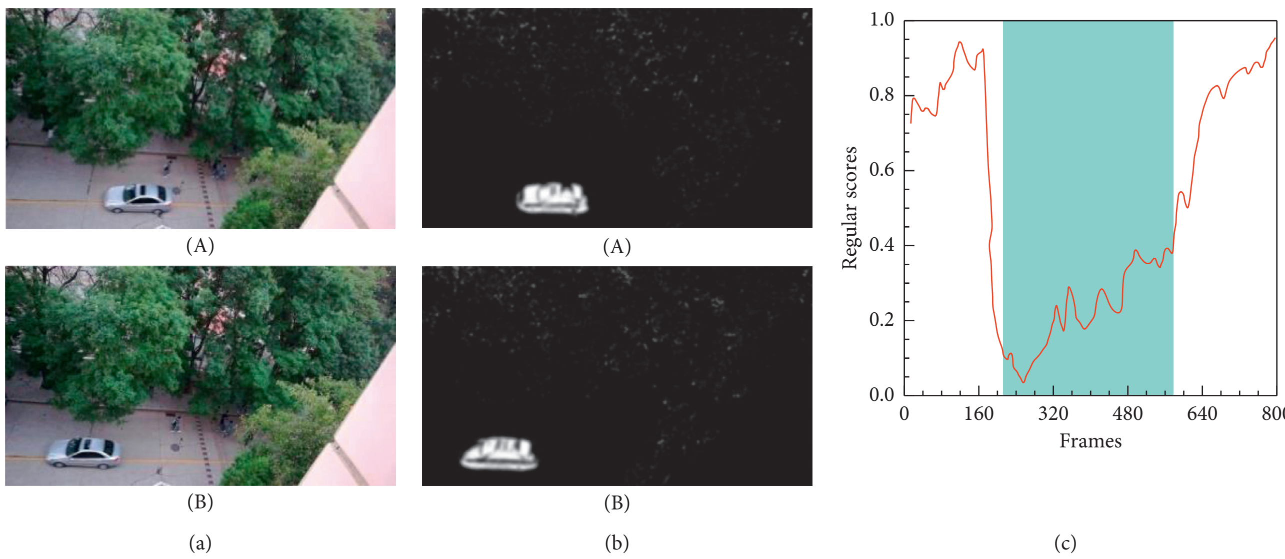


Figure 9: Frame-level evaluation results on the recorded real video. (a) Ground truth of the 432th frame (A) and 496th frame (B) labelled as abnormal. (b) Difference between the ground truth and the corresponding predicted frame. (c) Relationship between the recorded real video frames and the regular score.

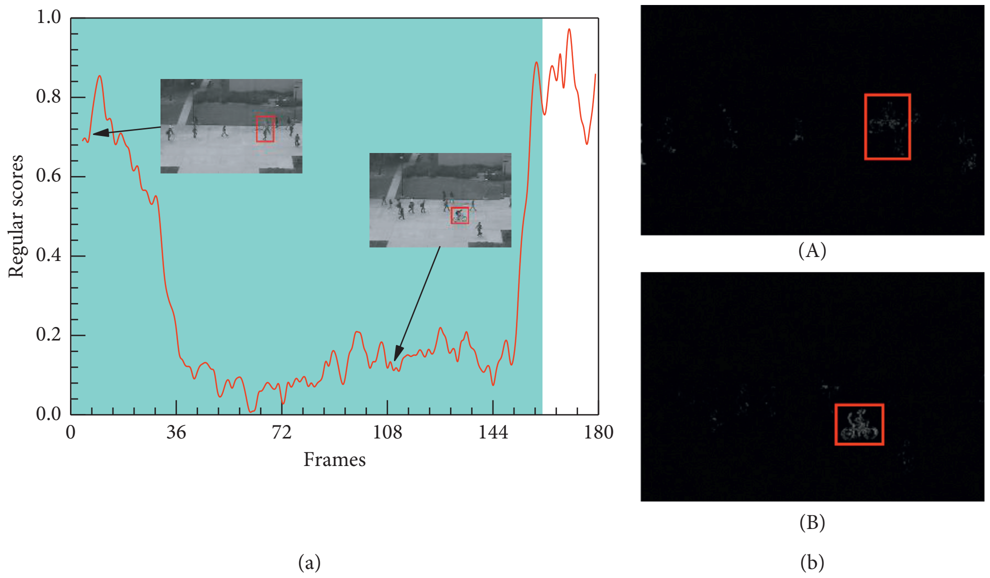


Figure 10: Frame-level evaluation results on Ped2 6th test video. (a) Relationship between the test video frames and the regular score. (b) Difference between the ground truth and the corresponding generated frame.

other methods except ConvLSTM [8] (Ped1) and AnomalyNet [37] (Ped2). The experimental results demonstrate the superiority of our approach in anomaly detection task.

Moreover, we choose the more typical per sample prediction time (i.e., average running time comprises the prediction frame generation and anomaly detection) to evaluate the complexity of the proposed solution. Table 4 shows the running time of our approach in comparison with several previous methods on UCSD Ped2 dataset. It can be seen that our method is a little bit slow than MDT [34] and Unmasking [36], but the AUC value obtained by our model is superior to these methods. Besides, our approach runs almost as fast as baseline [15]. The reason lies in that we add the HDC module and the DB-ConvLSTM module, which takes time. In general, our method can ensure running time and accuracy to be better working on a real world.

3.6. Ablation Studies. To verify the effectiveness of each component of the proposed method, we conducted an ablation study for different component. For comparison, three variants of the proposed method (i.e., STP-net only with

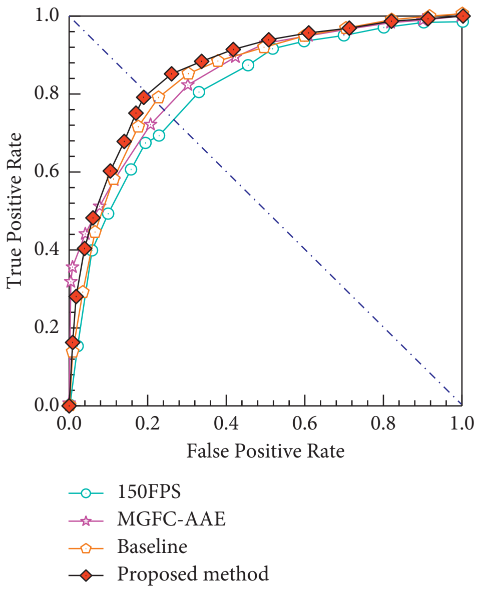


FIGURE 11: ROC curves comparison of different methods on Avenue dataset.

TABLE 1: Frame-level AUC performance of different methods on Avenue dataset.

Methods	AUC (%)
150FPS [33]	80.9
Conv-AE [7]	70.2
ConvLSTM [8]	80.3
ConvLSTM-AE [9]	77.0
MGFC-AAE [10]	84.2
Unmasking [36]	80.6
AnomalyNet [37]	86.1
Baseline [15]	84.9
Proposed method	86.4

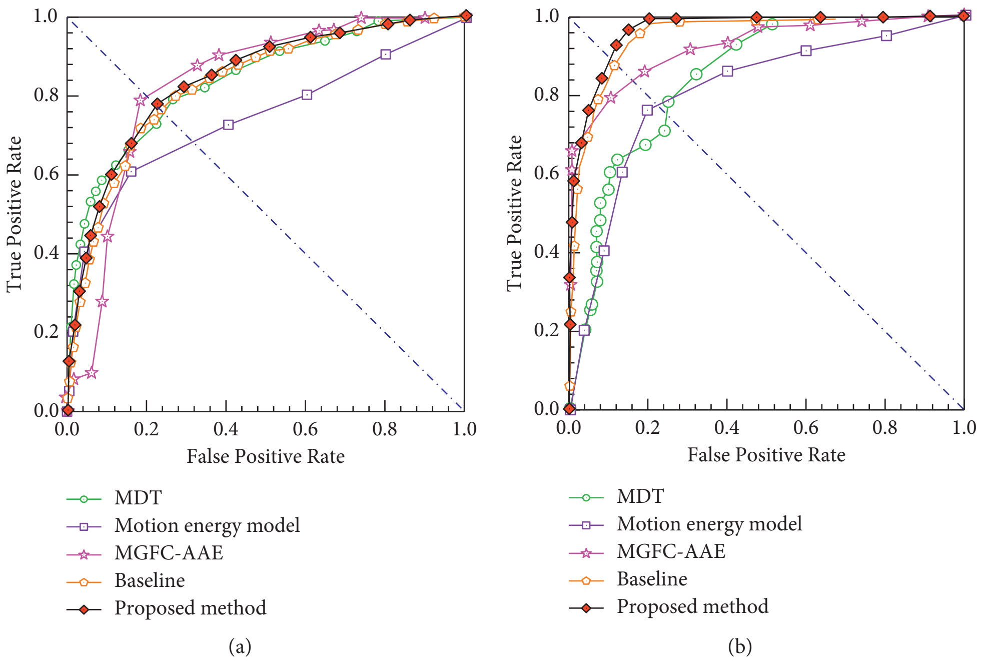


FIGURE 12: ROC curves comparison of different methods on UCSD dataset. (a) Frame-level ROC on Ped1. (b) Frame-level ROC on Ped2.

Table 2: Frame-level AUC performance of different methods on UCSD dataset.

Methods	AUC (%)	
	UCSD Ped1	UCSD Ped2
MDT [34]	81.8	82.9
Motion energy model [38]	75	81
Conv-AE [7]	81.0	90.0
ConvLSTM [8]	89.9	87.4
ConvLSTM-AE [9]	75.5	88.1
MGFC-AAE [10]	85	91.6
Unmasking [36]	68.4	82.2
AnomalyNet [37]	83.5	94.9
Baseline [15]	83.1	95.4
Proposed method	84.4	96.3

Table 3: Comparison of EER performance on different datasets.

Methods	EER (%)		
	UCSD Ped1	UCSD Ped2	Avenue
Conv-AE [7]	27.9	21.7	25.1
ConvLSTM [8]	12.5	12	20.7
MGFC-AAE [10]	20	16	22.3
AnomalyNet [37]	25.2	10.3	22
Baseline [15]	24	12	21
Proposed method	22.8	11	19.7

Table 4: Comparison of running time performance on UCSD Ped2 dataset.

Method	Running time (frames per second)
MDT [34]	23
Unmasking [36]	20
Baseline [15]	32
Proposed method	29

HDC, with ConvLSTM, and with DB-ConvLSTM) were trained to evaluate the performance for anomaly detection. Table 5 shows the AUC values obtained from the variants with different component on the different datasets. It can be observed that the variant with all components achieves the best results than those with fewer components, which shows the importance to take full advantage of the spatiotemporal features for anomaly detection. The HDC module can extract the more representative multiscale spatial features, and the DB-ConvLSTM module can memorize the temporal information. The experimental results indicate the effectiveness of our method, which fully considers spatiotemporal information.

In addition, we evaluated the effect of optical flow loss and RGB difference loss for our model on different datasets. As shown in Tables 6 and 7, when the RGB difference loss was employed in the network, the average runtime using batch data reduced from 0.4648 (s/batch) to 0.0036 (s/batch) and the AUC values significantly improve by 0.6% (Avenue), 0.5% (UCSD Ped1), and 0.6% (UCSD Ped2), respectively. It is obvious that the RGB difference loss replaced optical flow loss can greatly save the time of optical flow extraction and shorten the training time. In summary, our method gives a full consideration of the spatiotemporal information, thus effectively improving the accuracy of the detection results.

Table 5: Effect of different components on AUC values.

Components	AUC (%)		
	Avenue	UCSD Ped1	UCSD Ped2
HDC	85.4	83.8	95.7
ConvLSTM	85.2	83.5	95.4
DB-ConvLSTM	85.5	83.9	95.6
HDC and DB-ConvLSTM	86.4	84.4	96.3

Table 6: Effect of different type loss functions on runtimes.

Loss function	Running time (s/batch)		
	Avenue	UCSD Ped1	UCSD Ped2
With optical flow loss	0.4685	0.4643	0.4615
With RGB difference loss	0.0036	0.0036	0.0036

Table 7: Effect of different type motion loss function on AUC values.

Loss function	AUC (%)		
	Avenue	UCSD Ped1	UCSD Ped2
With optical flow loss	85.8	83.9	95.7
With RGB difference loss	86.4	84.4	96.3

4. Conclusions and Future Work

This paper proposes an effective anomaly-detection method based on the STP-net by integrating HDC and DB-ConvLSTM module. We employ the proposed network to capture more comprehensive multiscale spatial features and temporal information of regular events. In the testing stage, the abnormalities of the test video were detected by the lower regular scores calculated by the PSNR values between the predicted frames and actual frames. Furthermore, using RGB differences as motion loss can reduce the training time. To further evaluate the proposed model, we conducted a series of experiments on several public benchmark datasets. The experimental results show that the AUC values of the CUHK Avenue, UCSD Ped1, and Ped2 datasets reached 86.4%, 84.4%, and 96.3%, respectively. Our method performs well compared with the state-of-the-art approaches in terms of detection accuracy through qualitative analysis and quantitative comparisons.

The proposed method does not limit the type of abnormality, and it can achieve the general detection of different abnormal behaviors in a specific scenario. Therefore, our method can be conveniently applied to various video surveillance scenarios. However, this approach still has some shortcomings and limitations. First, the prediction method is highly dependent on prior information; thus, the detection results are sensitive to any changes of the previous frame. Second, our method might perform poorly on fairly easy to detect the abnormalities due to the occluded abnormal events. Third, the prediction network relies on the completeness of training data, implying that the training data should contain all normal behaviors of the scenario. To develop a complete anomaly detection system, as part of the future scope, we plan to exploit visual tracking technology to solve the problem of sensitivity and occlusion. Meanwhile, we will extend existing datasets to cover as many different surveillance video scenarios as possible to address smart-city and public-security issues.

Data Availability

The original data used to support the findings of this study are included in the article.

Conflicts of Interest

The authors declare that they have no potential conflicts of interest with respect to the research, authorship, and publication of this paper.

Acknowledgments

This work was sponsored by the National Natural Science Foundation of China (Grant nos. 61842103, 61871351, and 61801437), the Science and Technology Foundation of State Key Laboratory Of Electronic Testing Technology (Grant no. 6142001180410), and the Technological Innovation Foundation of the Higher Education Institutions of Shanxi Province, China (Grant nos. 2020L0301 and 2020L0389).

References

[1] Z. L. Zhao, “Community public safety evaluation system based on location information service architecture,” *Mobile Information Systems*, vol. 2021, Article ID 6694757, 10 pages, 2021.

[2] A. A. Sodemann, M. P. Ross, and B. J. Borghetti, “A review of anomaly detection in automated surveillance,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 1257–1272, 2012.

[3] F. Tung, J. S. Zelek, and D. A. Clausi, “Goal-based trajectory analysis for unusual behaviour detection in intelligent surveillance,” *Image and Vision Computing*, vol. 29, no. 4, pp. 230–240, 2011.

[4] D. Navneet and B. Triggs, “Histograms of oriented gradients for human detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 886–893, San Diego, CA, USA, June 2005.

[5] N. Dalal, B. Triggs, and C. Schmid, “Human detection using oriented histograms of flow and appearance,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 428–441, Berlin, Germany, May 2006.

[6] L. Kratz and K. Nishino, “Anomaly detection in extremely crowded scenes using spatio-temporal motion pattern models,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1446–1453, Miami, FL, USA, June 2009.

[7] M. Hasan, J. Choi, and J. Neumann, “Learning temporal regularity in video sequences,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 733–742, Las Vegas, NV, USA, June 2016.

[8] Y. S. Chong and Y. H. Tay, “Abnormal event detection in videos using spatiotemporal autoencoder,” in *Proceedings of the International Symposium on Neural Networks (ISNN)*, pp. 189–196, January 2017.

[9] W. Luo, W. Liu, and S. Gao, “Remembering history with convolutional lstm for anomaly detection,” in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME)*, pp. 439–444, Hong Kong, China, July 2017.

[10] N. Li and F. Chang, “Video anomaly detection and localization via multivariate Gaussian fully convolution adversarial autoencoder,” *Neurocomputing*, vol. 369, pp. 92–105, 2019.

[11] M. George, B. R. Jose, J. Mathew, and P. Kokare, “Autoencoder-based abnormal activity detection using parallelepiped spatio-temporal region,” *IET Computer Vision*, vol. 13, no. 1, pp. 23–30, 2019.

[12] A. Munawar, P. Vinayavekhin, and G. D. Magistris, “Spatio-temporal anomaly detection for industrial robots through prediction in unsupervised feature space,” in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1017–1025, Santa Rosa, CA, USA, March 2017.

[13] R. Villegas, J. Yang, and Y. Zou, “Learning to generate long-term future via hierarchical prediction,” in *Proceedings of the 34th International Conference on Machine Learning (ICML)*, vol. 70, pp. 3560–3569, Sydney, Australia, April 2017.

[14] Y. Zhao, B. Deng, and C. Shen, “Spatio-temporal autoencoder for video anomaly detection,” in *Proceedings of the 25th ACM International Conference on Multimedia*, pp. 1933–1941, Mountain View, CA, USA, October 2017.

[15] W. Liu, W. Luo, D. Lian, and S. Gao, “Future frame prediction for anomaly detection-a new baseline,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6536–6545, Salt Lake City, UT, USA, June 2018.

[16] C. Gao, S. Ye, H. Tian, and Y. Yan, “Multi-scale single-stage pose detection with adaptive sample training in the classroom scene,” *Knowledge-Based Systems*, vol. 222, Article ID 107008, 2021.

[17] S. Oh, S. Han, and J. Jeong, “Multi-scale convolutional recurrent neural network for bearing fault detection in noisy manufacturing environments,” *Applied Sciences-Basel*, vol. 11, no. 9, Article ID 3963, 2021.

[18] T. Akilan, Q. M. J. Wu, and W. Zhang, “Video foreground extraction using multi-view receptive field and encoder-decoder DCNN for traffic and surveillance applications,” *IEEE Transactions on Vehicular Technology*, vol. 68, no. 10, pp. 9478–9493, 2019.

[19] A. Shahbaz and K.-H. Jo, “Deep atrous spatial features-based supervised foreground detection algorithm for industrial surveillance systems,” *IEEE Transactions on Industrial Informatics*, vol. 17, no. 7, pp. 4818–4826, 2021.

[20] T. Ku, Q. Yang, and H. Zhang, “Multilevel feature fusion dilated convolutional network for semantic segmentation,” *International Journal of Advanced Robotic Systems*, vol. 18, no. 2, Article ID 17298814211007665, 2021.

[21] H. Song, W. Wang, and S. Zhao, “Pyramid dilated deeper convlstm for video salient object detection,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 715–731, Munich, Germany, September 2018.

[22] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[23] L. Wang, Y. Xiong, Z. Wang et al., “Temporal segment networks for action recognition in videos,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 11, pp. 2740–2755, 2019.

[24] A. Dosovitskiy, P. Fischer, and E. Ilg, “Flownet: learning optical flow with convolutional networks,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2758–2766, Santiago, Chile, December 2015.

[25] I. Goodfellow, J. Pouget-Abadie, M. Mirza et al., “Generative adversarial networks,” *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[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 (CVPR)*, pp. 1125–1134, Honolulu, HI, USA, July 2017.

[27] X. Shi, Z. Chen, H. Wang, and D. Yeung, “Convolutional lstm network: a machine learning approach for precipitation nowcasting,” 2015, https://arxiv.org/abs/1506.04214v1.

[28] Z. Cui, R. Ke, Z. Pu, and Y. Wang, “Stacked bidirectional and unidirectional LSTM recurrent neural network for forecasting network-wide traffic state with missing values,” *Transportation Research Part C: Emerging Technologies*, vol. 118, Article ID 102674, 2020.

[29] Y. Li, Y. Cai, J. Liu, S. Lang, and X. Zhang, “Spatio-temporal unity networking for video anomaly detection,” *IEEE Access*, vol. 7, pp. 172425–172432, 2019.

[30] L. Jin, F. Tan, and S. Jiang, “Generative adversarial network technologies and applications in computer vision,” *Computational Intelligence and Neuroscience*, vol. 2020, Article ID 1459107, 17 pages, 2020.

[31] Y. Tang, L. Zhao, S. Zhang, C. Gong, G. Li, and J. Yang, “Integrating prediction and reconstruction for anomaly detection,” *Pattern Recognition Letters*, vol. 129, pp. 123–130, 2020.

[32] Z. Wang and A. C. Bovik, “Mean squared error: love it or leave it? A new look at Signal Fidelity Measures,” *IEEE Signal Processing Magazine*, vol. 26, no. 1, pp. 98–117, 2009.

[33] C. Lu, J. Shi, and J. Jia, “Abnormal event detection at 150 fps in MatLab,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2720–2727, Sydney, NSW, Australia, December 2013.

[34] V. Mahadevan, W. Li, V. Bhalodia, and N. Vasconcelos, “Anomaly detection in crowded scenes,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, CA, USA, June 2010.

[35] D. P. Kingma and J. L. Ba, “Adam: a method for stochastic optimization,” 2015, http://de.arxiv.org/pdf/1412.6980.

[36] R. T. Ionescu, S. Smeureanu, B. Alexe, and M. Popescu, “Unmasking the abnormal events in video,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2895–2903, Venice, Italy, October 2017.

[37] J. T. Zhou, J. Du, H. Zhu, X. Peng, Y. Liu, and R. S. M. Goh, “Anomalynet: an anomaly detection network for video surveillance,” *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 10, pp. 2537–2550, 2019.

[38] T. Chen, C. Hou, Z. Wang, and H. Chen, “Anomaly detection in crowded scenes using motion energy model,” *Multimedia Tools and Applications*, vol. 77, no. 11, pp. 14137–14152, 2018.

Hindawi
Computational Intelligence and Neuroscience
Volume 2021, Article ID 3110416, 12 pages
https://doi.org/10.1155/2021/3110416

Research Article

A Smart Surveillance System for Uncooperative Gait Recognition Using Cycle Consistent Generative Adversarial Networks (CCGANs)

Wafaa Adnan Alsaggaf[1] **Irfan Mehmood**[2] **Enas Fawai Khairullah**[1] **Samar Alhuraiji**[3] **Maha Farouk S. Sabir**[4] **Ahmed S. Alghamdi**[5] **and Ahmed A. Abd El-Latif**[6]

[1]*Department of Information Technology, Faculty of Computing and Information Technology King Abdulaziz University, Jeddah 23713, Saudi Arabia*
[2]*Faculty of Engineering & Informatics, School of Media, Design and Technology, University of Bradford, Bradford, UK*
[3]*Department of Computer Science, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia*
[4]*Department of Information Systems, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia*
[5]*Department of Cybersecurity, Faculty of Computer Science and Engineering, University of Jeddah, Jeddah, Saudi Arabia*
[6]*Department of Mathematics and Computer Science, Faculty of Science, Menoufia University, Shibin Al Kawm 32511, Egypt*

Correspondence should be addressed to Ahmed A. Abd El-Latif; a.rahiem@gmail.com

Received 15 June 2021; Revised 13 September 2021; Accepted 29 September 2021; Published 13 October 2021

Academic Editor: Francisco Gomez-Donoso

Surveillance remains an important research area, and it has many applications. Smart surveillance requires a high level of accuracy even when persons are uncooperative. Gait Recognition is the study of recognizing people by the way they walk even when they are unwilling to cooperate. It is another form of a behavioral biometric system in which unique attributes of an individual's gait are analyzed to determine their identity. On the other hand, one of the big limitations of the gait recognition system is uncooperative environments in which both gallery and probe sets are made under different and unknown walking conditions. In order to tackle this problem, we propose a deep learning-based method that is trained on individuals with the normal walking condition, and to deal with an uncooperative environment and recognize the individual with any dynamic walking conditions, a cycle consistent generative adversarial network is used. This method translates a GEI disturbed from different covariate factors to a normal GEI. It works like unsupervised learning, and during its training, a GEI disrupts from different covariate factors of each individual and acts as a source domain while the normal walking conditions of individuals are our target domain to which translation is required. The cycle consistent GANs automatically find an individual pair with the help of the Cycle Loss function and generate the required GEI, which is tested by the CNN model to predict the person ID. The proposed system is evaluated over a publicly available data set named CASIA-B, and it achieved excellent results. Moreover, this system can be implemented in sensitive areas, like banks, seminar halls (events), airports, embassies, shopping malls, police stations, military areas, and other public service areas for security purposes.

1. Introduction

Biometric systems employ the human unique characteristics that are either physical or behavioral to determine their identity. All of these characteristics distinguish one individual from the other [1]. Similarly, a Gait Recognition system is also a biometric system that identifies an individual based on the way they walk. It is a relatively new technology that has attracted a large number of researchers in recent years. Previously developed biometric systems, such as face recognition, iris recognition, and fingerprint recognition, used physical attributes of individuals to establish their identity, but gait recognition examines individual's behavior to identify them, making it a behavioral biometric system [2]. The advantages of gait recognition biometric over visual biometrics include that gait recognition is appropriate and effective in recognizing people even when their faces are covered and also when the distance between surveillance cameras and individuals is about 50 meters. On the other hand, iris recognition requires that an individual must be at a distance of 3 cm from the camera, and face recognition requires about 5 meters. Besides this, visual biometrics also requires high-quality images for accurate identification and requires the involvement of the subject during identification. On the other hand, gait recognition is such an admirable biometric system that does not require subject cooperation at the time of inspection from surveillance cameras, and this aspect makes it effective in ensuring security at public safety areas [3].

Different researchers carried out gait recognition in two different experimental settings. One is the cooperative manner, and other is the uncooperative environment. In a cooperative manner, the model is trained and tested on known walking conditions [4]. More specifically, if a model is trained on several individuals who walk normally in a video without carrying any objects or items and then tested on different videos of individuals but with the same walking condition, i.e., normal walk. In this case, the walking condition of the individual being examined is known to the model during validation. On the other hand, in an uncooperative manner, the individual walking condition is kept hidden from the model. The model is trained on a set of individuals who walk with normal style and tested on same individuals but with different walking styles such as a walk with carrying items, such as bags and suitcases in their hands, or with varied clothing conditions such as coats and jackets. These items, or varied walking conditions, are referred to as covariate factors, and they cause individual gait characteristics to be disrupted. Moreover, the second experimental setting is more realistic and very challenging task in computer vision because the performance of the gait recognition model drastically drops when individuals come with unknown and variable covariate conditions. All these are dynamic walking conditions and strongly affect the recognition rate of identifying individuals from their gait. In existing works, when the subjects are considered as cooperative, that is, covariate conditions are known during the training of the system, then the model or system reveals a very superior performance overall. However, there is a drop in outcomes in an experimental scenario when uncooperative environments are considered, and gallery and probe sets are formed under unknown walking conditions.

In this research, we attempt to improve the recognition performance of the deep learning model with the presence of these covariate factors. So in order to improve the results and make the system strong to any dynamic change, we employ cycle consistent generative adversarial networks (CCGANs) [5]. The main objective of this research is to train a model with one walking condition of an individual that is normal walk and validate and test the model with unknown and dynamic walking conditions, i.e., what if a person comes with a bag or a coat and any other dynamic real-time condition. We first utilized a deep CNN model, which is train on a unique gallery set on all individuals with normal walk style. In the second stage, we validated the model with individuals carrying a bag or wearing coats or any other thing. The presence of these covariate factors modifies the gait features of individuals and is different from the features that are extracted by the CNN on a normal walk. So, before passing the tested image directly to trained CNN model, it first goes via CCGANs, which translates a Gait energy image (GEI [6], a more compact gait representation) disrupted with varied covariate factors to a regular walking GEI and recovers the gait features on which CNN is trained to recognize the person. CCGANs are trained in an unsupervised manner, that is, during translation of disrupted GEI to normal GEI, it automatically picks up the right normal GEI from the target domain for disrupted GEI using cycle loss. In addition, currently, different types of GANs are used for different types of tasks. For example, a very basic GAN [7] generates specific artificial images on which it was trained from a random vector, and there is no control over the data that are generated by the model. In our case, we do not want to generate artificial data from a random vector; therefore, basic types of GANs are not applicable. Similarly, conditional GANs [8] are used when we want to convert an image from one domain to an image from a different domain. In this case, we must have paired data, for example, if we want to translate map photographs to aerial photographs, then we must specify that a given instance of the map image is translated to a specific image of aerial photograph during training. In the presented problem, the disrupted probe set needs to be translated into a normal probe set. But the probe set contains images of various individuals, and it would be unjust to indicate at the testing time that a certain GEI of a person walking with bags, such as Person-ID-001, is translated into Person-ID-001 with a normal walk. Because at testing time, we do not know the person's actual label or ID. So, conditional GANs are also not suitable for this problem. However, CCGANs do not have such limitations. So, the main reason for employing CCGANs is that we want to translate the images affected from covariate factors to normal GEIs without revealing the IDs of the persons. CCGANs' unsupervised mechanism is a perfect match for the solution to the challenge at hand because it automatically finds the exact pair for a particular person followed by a translation of the image. The main contributions of this work are as follows:

(i) Handling the unknown and dynamic walking environment using CCGANs.

(ii) Reconstruction of gait features, which are lost by variable covariate factors.

(iii) CCGANs are not previously used for gait analysis to recover gait features. The experimental findings demonstrate the capability of CCGANs in recovering gait features of individuals for an uncooperative environment and give better performance.

(iv) Comparative analysis of our approach with other methods shows the superiority of the proposed method.

The rest of the article is organized into several sections. Section 2 presents the related work, Section 3 describes the proposed methodology, and Section 4 presents the results followed by conclusion.

2. Literature Review

Currently, gait recognition methods mainly fall into two main approaches: one is model-based approaches [9–11] and the other is appearance-based approaches [12–14]. In model-based approaches, the human body structure is focused to extract gait features with the use of different body parameters, including stride, speed, size of different body parts, and all other static and dynamic parameters. This type of approach is computationally expensive because it requires high-resolution images, even though the gait recognition method is supposed to work with low-quality images and with a minimum amount of light. On the other hand, the appearance-based approaches work on the human silhouettes extracted from different sequences, and mainly, the motion of the human body is focused in this approach. This type of method can work with low-quality images, which are appropriate for real-time public surveillance. The appearance-based approaches are further divided into hand-crafted feature extraction-based methods and deep learning-based methods.

In the context of traditional machine learning methods, Anusha and Jaidhar [12] extract the hand-crafted features from the selective regions of Gait energy image (GEI) [15], which is a popular gait representation using Modified Local Optimal Oriented Pattern (MLOOP) feature descriptor followed by dimensionality reduction algorithm to reduce extracted feature vectors, and then, classification is performed. The proposed approach is validated on the CASIA-B and OU-ISIR B gait data sets, and it works brilliantly in all experiments. Similarly, Lishani et al. [16] also employed the traditional machine learning technique for gait recognition. In this work, Multiscale Local Binary Pattern (MLBP) and Gabor filter bank feature descriptor are used to extract features followed by Spectra Regression Kernel Discriminant Analysis (SRKDA) algorithm for feature selection. In the end, the *K*-nearest neighbor classifier is utilized for classification and achieved the recognition score of 92%. Rokanujjaman et al. [17] introduced a new gait representation termed as frequency-domain gait entropy (EnDFT), from which features are taken from the less affected part of EnDFT, and a distance metric is used to distinguish humans. Similarly, Bashir et al. [13] select the pairwise features using their proposed gait entropy image (GenI) and Adaptive Component and Discriminant Analysis (ACDA). Moreover, Gupta et al. [18] proposed boundary Energy Image (BEI) based gait representation in which contours of all silhouette images of humans are averaged. They use Principal Component Analysis (PCA) for dimensionality reduction and Linear Discriminant Analysis (LDA) for classification and attained encouraging performance.

Currently, deep learning-based methods show exceptional performances in computer vision-related tasks in various domains [19–22]. Similarly, for gait recognition, it is also proved to be an excellent approach in the recognition of individuals from their gait characteristics. Alotaibi and Mahmood [14] proposed a deep specialized convolutional neural network (CNN) with eight layers to classify the subjects. The proposed model is validated on the CASIA-B gait data set with different experimental settings. Hawas et al. [23] proposed the CNNs with the optical flow of GEI to increase the performance of the model. The optical flow excludes the static part of GEI having high pixel intensities and represents only the dynamic part with high intensities of GEI. Similarly, Linda et al. [24] proposed color-mapped contour gait images (CCGIs) and deep CNN for cross-view gait recognition. CCGIs is helpful in discriminating temporal information in human walking sequences. The model is evaluated on CASIA-B gait data set and produces an average accuracy score of 94.65%. Su et al. [25] introduced center ranked loss in their deep neural network to integrate information of all positive and negative samples. Huang et al. [26] proposed the gait recognition model based on keyframes with deep learning techniques. The total frames in a sequence have different contributions towards gait characteristics. For this, they proposed an extraction module that extracts the keyframes from a given gait sequence and results in the extraction of highly distinctive features. Yao et al. [27] proposed a skeleton gait energy image (SGEI) using multibranch CNNs, in which one branch is responsible for predicting confidence maps, and the second branch is used to predict the affinity fields. After that, the results and features of the image from both branches are concatenated to perform gait recognition. Ling et al. [28] employed the attention mechanism-based approach in gait recognition to emphasize discriminating regions. They validate their approach on OU-ISIR TREADMILL data set B to handle the problem of covariate conditions. Moreover, the use of transfer learning is also employed to gait recognition in which Wu et al. [29] proposed the DenseNet-based transfer learning method. The spatial information of gait from GEI is given as an input to DenseNet to extract features of each subject, and finally, the KNN is used to classify individuals respectively.

Furthermore, in some recent research studies, the study of human behavior in different domains is also exploited. In the context of activity recognition, Shu et al. [30] address the problem of group activity recognition in the multiple-person scene. They proposed the Graph LSTM-in-LSTM (GLIL) based framework, which simultaneously models both individual- and group-level activities. Similarly, Tang et al. [31] employed the motion characteristics of humans to solve the problem of group activity recognition. Relevant motions of individuals are captured by the constraint of Context Coherence (STCC) and a Global Context Coherence (GCC). Kabir et al. [32] recognized the activity of humans employing state-space linear modeling. A coefficient matrix is used to define the association between states and inputs. Furthermore, the most advanced research by Shu et al. [33] aimed to predict future motions of individuals based on currently observed motions. Moreover, the limitations of a different algorithm for activity recognition are also well researched. Choe et al. [34] showed the high operation complexity of the KNN algorithm for activity recognition and proposed a method to reduce this complexity. Besides the gait recognition in the field of biometrics, there also exists some other possible scenarios and application areas, such as Big Data and other IoT systems, in which similar kinds of work can be applied. Content analysis of videos is also used in other fields. For example, Song et al. [35] analyzed and reviewed the latest methods on the content analysis of the videos for action recognition. Instead of action recognition, the behaviors of humans are also used for the gesture, speech, and wrist-activity recognition. For example, Jo et al. [36] proposed the novel method for Hidden Markov Model (HMM) based speech recognition. Their approach is based on a modified version of the Viterbi scoring method. In order to determine an optimal matching model, Viterbi scoring plays a remarkable role in speech recognition systems based on HMM. A dynamic recognition system based on human gestures is introduced by Chen et al. [37], which are applicable in IPTV remote control. A hardware accelerator is also designed for object detection of real-time moving objects. These surveillance systems captured the videos from several cameras, and these videos often suffer from various noises. Different researchers also proposed different techniques to overcome the noise problem. For example, Niu et al. [38] proposed an approach to remove blurriness in images of surveillance systems videos during a raining environment. Wang et al. [39] proposed a novel method of noise processing for underwater targets. This approach is further integrated with CNNs to identify the underwater target.

Currently, the advancements in computer vision-based algorithms and techniques are increasing rapidly and adopted for many daily life use cases. One of the most advanced algorithm is generative adversarial networks [7]. They are generative models based on deep learning methodologies and are categorized into supervised and unsupervised models. Cycle consistent generative adversarial network (CCGAN) is one of the types of unsupervised generative models, which performs image translation in unsupervised manner [5]. It is used in most of the computer vision-related domains. Recently, Kearney et al. [40] involved the attention mechanism in CCGANs to perform image translation of MRI to CT scans. Similarly, Armanious et al. [41] translated the Positron Emission-computed Tomography (PET) images to CT scan images using CCGANs along with nonadversarial cycle losses. On the other hand, when it comes to gait recognition, Yu et al. [42] proposed a framework, namely, GaitGAN to select invariant features of an individual's gait to reduce the effect of covariate factors. However, the presented approach using GANs model is trained in the supervised manner in which the source and target GEIs are already known. Similarly, for view-specific gait recognition, He et al. [43] employed the multitask GANs to directly change view specific attributes in the latent space. Furthermore, the alpha-blending GANs are employed by Li et al. [44] to translate the GEIs affected by carried objects to GEIs without carried objects. It is reasonable to draw a conclusion that analysis of human behavior for different kinds of tasks, such as person identification for the biometric systems, speech recognition, gesture recognition, and action recognition, are active research areas in the current era.

3. Proposed Methodology

The proposed methodology of the gait recognition system is presented in Figure 1. As shown in Figure 1, we first train the CNN model on a gallery set, which is composed of individual's GEIs during a normal walk. Later, we used the CCGAN model to convert the image disrupts from covariate factors to regular GEIs and make recognition possible and accurate in the uncooperative setting of experimentation.

3.1. Input Data.

3.1. Input Data. The popular gait representation, namely, gait energy image (GEI) is used as an input to the proposed approach [6]. It can be computed by first extracting the frames of a sequence of an individual followed by computation of silhouettes using background subtraction from each frame. Then, at the last, all the silhouette images of one sequence of a particular individual are averaged and aligned to form a gait energy image representation. Equation (1) shows the mathematical formulation of GEI:

$$\mathrm{GEI} = G(x, y) = \frac{1}{T}\sum_{t=1}^{T} I(x, y, t). \tag{1}$$

In equation (1), the $G(x, y)$ represents the resulting GEI, where $I(x, y, t)$ denotes the silhouette image with frame number t and coordinates x, y, while a total number of frames are represented by T. The main advantage of this gait representation is less storage space and computational time. The silhouette images are prone to noise, and processing each silhouette frame is too costly; therefore, GEIs are the most effective gait representation. Furthermore, the GEIs are also very sensitive to different covariate factors, which strongly affect the shape of the GEI. Some examples of gait energy images of the CASIA-B data set with different walking conditions are shown in Figure 2.

3.2. CNN Architecture. After extracting the GEIs of all subjects, we resized and rescaled them to $240 \times 240 \times 1$ image dimension and give them as an input to CNN architecture. We have used the same CNN architecture proposed by Bukhari et al. [4]. In this CNN architecture, there are four convolution blocks. Each block is composed of 3×3 convolution layers, followed by 2×2 max-pooling layers to reduce the spatial dimensions of the input data and select the maximum value from the input region as per the following equation:

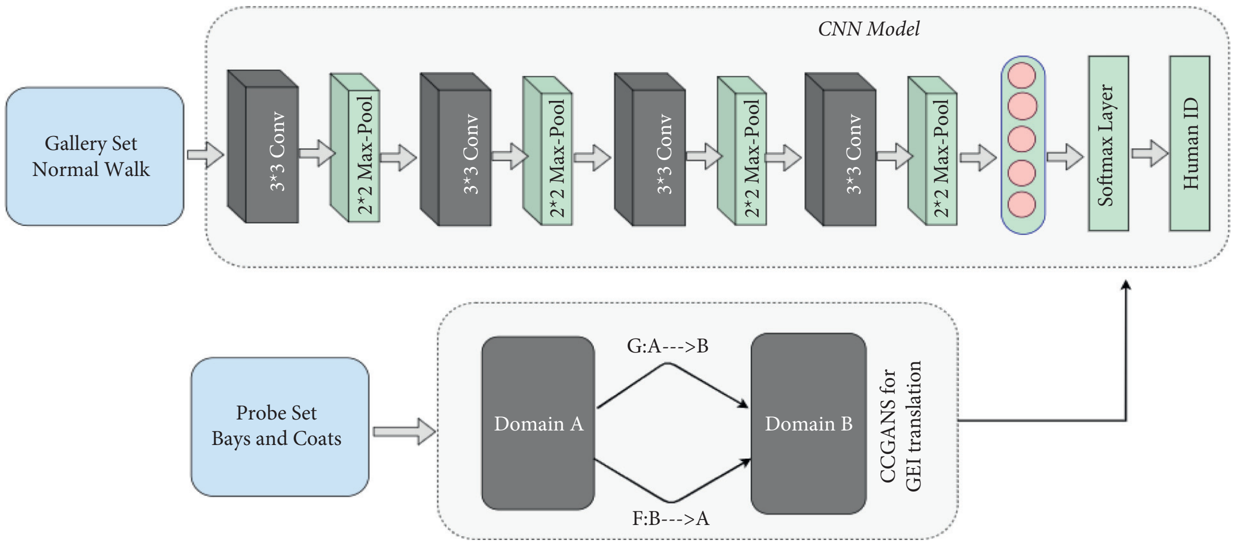


Figure 1: The proposed framework for gait recognition in uncooperative environment.

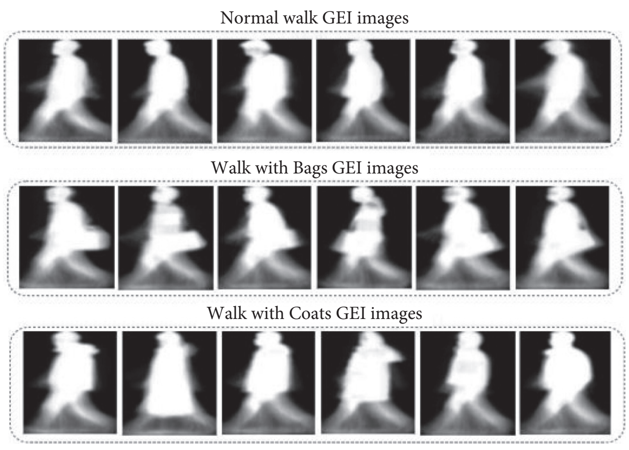


Figure 2: Examples of CASIA-B gait data set in which the first row corresponds to GEIs with a normal walk, the second rows correspond to subjects carrying a bag, and the third row corresponds to subjects wearing coats.

$$y^i_{k.w} = \max_{0\le a,b\le p}\left(xi_{k\times p+a,w\times p+b}\right). \tag{2}$$

In the above equation, a neuron or unit $y^i_{k.w}$ on down sampling layer is present on a particular position (k, w) in i^{th} output map. On the region $p\times p$, a maximum value is chosen and assigned to $y^i_{k.w}$ neuron in the i^{th} input map x_i. Subsequently, the activation function used after each convolution layer is Leaky ReLu to achieve nonlinearity. It is defined in the following equation:

$$f(x) = \left\{\begin{array}{cc} x & \text{if } x > 0 \\ 0.01x & \text{otherwise} \end{array}\right\}. \tag{3}$$

In the above equation, x is a feature map resulted from the convolution layer. If the values in x are greater than zero, they are preserved; otherwise, they are multiplied by a small value of 0.01. This activation function is an improved form of the ReLu activation. In the case of ReLu activation function, the gradient is 0 for all input values less than zero, deactivating the neurons in that region and perhaps causing the dying ReLu problem. On the other hand, Leaky ReLu overcomes this problem. In the case of Leaky ReLu, instead of converting the gradient 0 for all negative values, it returns a small number multiplied by 0.01 times x. This small value shows the influence of negative values in feature maps x. The higher the value, the lower the influence. As a consequence, it also produces outputs for negative values. Furthermore, all the convolutions are not padded. The feature maps, which are the output of the convolutional layers, can be computed by the following equation:

$$X = b_i \sum_{k=1}^{n} W_{i,j} * Y. \tag{4}$$

In the above equation, X is the input to the layer and Y is the output of layer in which $*$ denotes the operation of convolution, where b_i is bias term and $W_{ij}\,(i, j \in N)$ is weight optimized by weight optimizer algorithm, and size of input data is denoted by n. Moreover, all the convolution blocks in our proposed architecture are responsible for extracting the semantic information from the GEI and make a feature set that discriminates each individual based on their gait. At the last, two fully connected layers are used. The number of neurons in the last layer is equal to a number of class labels or individuals present in the data set followed by the "softmax" activation function, which returns the probabilities of every class.

The architecture is trained on 124 individuals present in the CASIA-B gait data set with normal walking conditions. In addition, the epochs are set to 30 with weight optimizer Adam along with a learning rate of 0.0001, respectively.

3.3. *Cycle Consistent Generative Adversarial Networks (CCGANs) for Reconstruction of Gait Features*

3.3.1. Problem Formulation. To satisfy the contribution of the presented work, we have employed the cycle consistent generative adversarial network (CCGANs). The main objective of this study is to handle the unknown and dynamic walking conditions. More specifically, we seek to reconstruct the gait features, which are destroyed by the presence of covariate factors. So, after compiling the input data in the form of GEIs for each subject, we created the CNN model. This CNN model is trained on normal walking sequences (videos) of all subjects, that is, 124. Now, if one of those 124 individuals comes in front of a surveillance camera while carrying a bag for identification, then the system performs poorly due to the occlusion of the bags, which disrupts the gait features. Because when a bag is placed in front of certain areas of the body, they become hidden, and hence, important information regarding the gait style of an individual is also lost. The same is the case with other carrying items and wearing conditions. Therefore, it is difficult to identify the person. We may also consider it as how to recognize the individuals from their faces if faces are covered by masks because masks are occlusions that hide the features of the face. So, a cycle consistent GAN model is deployed to recover/reconstruct the original image or gait features that are affected by these covariate factors. This model was proposed by Zhu et al. for image-to-image translation tasks when the source and target images are not paired [5]. The advantage of this model is that it carries out training without paired images. In our scenario, we reconstruct the GEIs of bags and coats of each individual to a corresponding normal GEI so that features are recovered and human recognition is possible and CNN is able to predict the subject ID with more accurate results. Furthermore, CCGANs efficiently handle dynamic covariate conditions of bags and coats that are unknown to the model; that is, we train the CNN model on individuals with normal walk styles and validate it on individuals' walks while carrying bags and wearing coats. Furthermore, Table 1 shows the symbol to describe all used notations or keywords.

3.3.2. Architecture and Design Mechanism of CCGANs. The model architecture of Cycle Consistent GANs is composed of two generator models and two discriminator models. The generator models are responsible for generating the required images, and discriminator models are used to discriminate between the real and fake images generated artificially. Our source domain contains GEIs of bags and coats (Domain A), whereas the target domain contains the normal walk GEIs of individuals (Domain B). The input of the first generator (generator A) is the images from Domain B, that is, GEIs with a normal walk, and the output is the translated images to Domain A, which is bags and coats GEIs. Similarly, the input of the second generator (generator B) is the images from Domain A, and the output is translated images from Domain B. Each generator model is associated with its corresponding discriminator model. The discriminator A takes input images of bags and coats, which are domain A images along with the artificially generated images from generator A and predicts that whether the images are real or fake. Similarly, the second discriminator B takes normal real GEIs and artificially generated normal GEIs from generator B and then predicts that whether they are real or fake. The adversarial zero-sum loss is used to train both generator and discriminator models. The main objective of the generator model is to best fool the discriminator model with the help of generated images, and the discriminators models are learned to predict and detect the fake and real images of both domains. Furthermore, the CCGAN models use the term cycle because a cycle is created in the whole process. The input of generator B is the images from Domain A, that is, bags and coats, and the output of generator B is the normal GEIs, which are our required translation. This output is the input of the generator A to generate images of domain A and hence the cycle is created. Similarly, the identical procedure is followed with generator A.

Table 1: Notations and definitions.

Notations	Definitions
$\mathbf{D_A}$	Discriminator model of domain A (probe bags/coats)
$\mathbf{D_B}$	Discriminator model of domain B (normal GEIs)
$\mathbf{A}$	Training samples of domain A $\{a_i\}_{i=1}^{N} \in A$
$\mathbf{B}$	Training samples of domain B $\{b_i\}_{i=1}^{N} \in B$
$\mathbf{F}\{(\mathbf{b})\}$	Translated normal GEIs
$\mathbf{L_{cyc}}$	Cycle consistency loss function
$\mathbf{L_{GAN}}$	Adversarial loss function
$\mathbf{G}$	Translator of domain A
$\mathbf{F}$	Translator of domain B
min	Minimizing the variable
max	Maximizing the variable

The architectural configurations of the discriminator and generator model are the same as described in the previous literature [5]. The discriminator is simply a CNN that performs image classification and predicts that the image is real or fake. A convolution operation, followed by instance normalization instead of batch normalization is used in the discriminator model of the CCGANs. The activation function used in the whole discriminator model is Leaky ReLu. All the convolutions are padded convolution with a kernel size of $4 * 4$. Furthermore, the hyperparameters of the discriminator model include loss function, which is Mean

Squared Loss (MSE), and Adam optimizer used for optimizing weights with a learning rate of 0.0002. The formula of computing MSE loss during GEI translation is given by the following equation:

$$\mathrm{MSE} = \frac{1}{n} \cdot \sum_{i=1}^{n} \left(y_i - \hat{y}_i\right)^2, \tag{5}$$

where y_i and $\hat{y}_i$ are the actual and predicted class labels of the model and n is the total number of classes. On other hand, the architecture of the generator model is based on an encoder-decoder structure. The generator model first downsamples the image to get the context of the image up to the bottleneck layer and then encodes this context with the help of ResNet layers that uses the skip connections followed by upsampling layers to decode the context to the required output image.

3.3.3. Training Mechanism of CCGANs. The training of discriminator models consists of real and artificially generated images while the generator models are trained with the help of their discriminator models. Generator models use the adversarial loss and update and minimize it from the prediction of the discriminator as "real" for generated images. This encourages generator models to generate images that are closer to our required target domain images, i.e., normal GEIs. Moreover, the other loss functions that are used by the generator model include identity loss, forward loss, and backward loss. For the adversarial loss, consider the mapping function for bags/coats to normal GEI translation as $G\colon A \longrightarrow B$ and discriminator B of domain D_B, i.e., normal GEIs, then the objective is expressed in the following equation:

$$\begin{aligned} L_{\mathrm{GAN}}(G, D_B, A, B) = {} & \mathbb{E}_{\mathbf{b \sim p\ da\ ta(b)}}\left[\log D_B(b)\right] \\ & + \mathbb{E}_{\mathbf{a \sim p\ da\ ta(a)}}\left[\log\left(1 - D_B(G(a))\right)\right], \end{aligned} \tag{6}$$

where G is a generator that learns to generate the required images $G(a)$ that look closer to images of domain B, i.e., images with recovered gait features, while the objective of the discriminator D_B is to distinguish the generated images $G(a)$ and real images b. G is attempting to minimize this objective, whereas adversary D is seeking to maximize it.

$$\min G \max D_B L_{\mathrm{GAN}}(G, D_B, A, B). \tag{7}$$

Similarly, for mapping function $F\colon B \longrightarrow A$, the similar adversarial loss is introduced along with discriminator D_A and computed by the following equation:

$$\min F \max D_A L_{\mathrm{GAN}}(F, D_A, B, A). \tag{8}$$

Moreover, the forward and backward cycle consistency are also computed. Consider an image a from the domain A for which the reconstruction of gait features is needed. So, for the forward cycle consistency, the whole cycle in CCGAN image translation can bring an image a to its original form, i.e., $a \longrightarrow G(a) \longrightarrow F(G(a)) \approx a$. Similarly, for backward cycle consistency, we have instance b from the domain B, and G and F should also satisfy the equation of backward cycle consistency: $b \longrightarrow F(b) \longrightarrow G(F(b)) \approx b$. So, all the mechanism of cycle consistency loss is given in the following equation:

$$\begin{aligned} L_{\mathrm{cyc}}(G, F) = {} & \mathbb{E}_{\mathbf{a \sim p\ da\ ta(a)}}\left[\|F(G(a)) - a\|_1\right] \\ & + \mathbb{E}_{\mathbf{b \sim p\ da\ ta(b)}}\left[\|F(G(b)) - b\|_1\right]. \end{aligned} \tag{9}$$

Furthermore, the full objective is given by the following equation:

$$\begin{aligned} L(G, F, D_A, D_B) = {} & L_{\mathrm{GAN}}(G, D_B, A, B) \\ & + L_{\mathrm{GAN}}(F, D_A, B, A) + \lambda L_{\mathrm{cyc}}(G, F). \end{aligned} \tag{10}$$

In the above equation, the relative importance of two objectives can be controlled by λ. The main aim is to solve the objective given in equation (10):

$$G^*, F^* = \arg\min_{G,F,D_a,D_b} \max L(G, F, D_A, D_B). \tag{11}$$

It is observed that this model can be viewed as training of two "autoencoders" [45]. One auto encoder $F \circ G\colon X \longrightarrow X$ learned jointly with the second autoencoder $G \circ F\colon Y \longrightarrow Y$. However, both autoencoders have their internal structures, and an intermediate representation is used to map an image to itself that is a translation of image from one domain to another domain. These scenarios can be observed as a special case in the work of "adversarial autoencoders" [46], in which the bottleneck layer of an autoencoder is trained with the help of adversarial loss to generate any arbitrary target. In our case, the domain B, which contains normal GEIs of different subjects, is the target distribution for the autoencoder $A \longrightarrow A$.

4. Experimental Setup and Results

4.1. Data Set. In our proposed work, we have used a popular data set for gait recognition named CASIA Gait data set provided by the Chinese Academy of Sciences (CASIA) [47]. It is composed of three main parts named CASIA A, B, and C. For this research, CASIA B is considered. CASIA B is a large multiview gait data set with viewpoints starting from 0 to 180 degrees. For each viewing angle, it consists of data of 124 subjects with each subject having a total of ten sequences available. Of ten sequences, six sequences are normal walk sequences [nm-01 to nm-06], two sequences in which subjects are walking with a bag [bg-01, bg-02], and two sequences in which clothing conditions are focused and subjects walk with wearing a coat [cl-01, cl-02].

4.2. Evaluation Metrics. To evaluate the proposed model, the performance measure includes the accuracy, F1score, precision, and recall are considered [48]. The detail of each measure is given below:

4.2.1. Accuracy. The total number of true predictions by the underlying model from overall predictions is measured by accuracy, and it is defined in the following equation:

Table 2: Results of CNN and CCGANs.

Sr. no.	Probe set	Accuracy (%)	Precision	Recall	F1score
1	Normal (CNN)	97.98	0.98	0.97	0.97
2	Bags (CNN)	38	0.29	0.38	0.31
3	Coats (CNN)	24.19	0.17	0.24	0.19
5	Bags (CNN + CCGANs)	**79**	**0.72**	**0.79**	**0.74**
6	Coats (CNN + CCGANs)	**51.8**	**0.38**	**0.51**	**0.41**

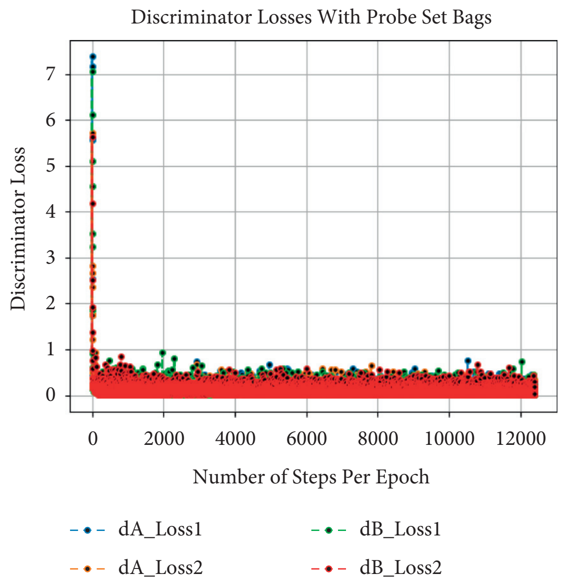


Figure 3: Graphs of different loss functions on both probe sets.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}. \tag{12}$$

4.2.2. Precision. Precision measures the total number of positive class predictions by the model that are in fact positive class predictions, and it is mathematically computed by the following equation:

$$\text{Precision} = \frac{TP}{TP + FP}. \tag{13}$$

4.2.3. Recall. Recall measures the total number of positive class predictions by the model from all positive cases, and it is mathematically computed by the folllowing equation:

$$\text{Recall} = \frac{TP}{TP + FN}. \tag{14}$$

4.2.4. F1 Score. The both precision and recall metrics are merged in F1 score to measure the entire performance of the model. Mathematically, it is described in the following equation:

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{15}$$

4.3. Results and Discussion. All the simulations with python implementation are run on Google Colab with a single 12 GB NVIDIA Tesla K80 GPU. It is necessary to mention here that in our whole experimental setup, we handle the unknown and dynamic walking conditions; that is, our gallery set consists of all 124 individuals with a normal walking condition with each individual having four sequences [nm-01-nm-04] as used by other researchers [13, 29, 49]. So, our CNN model is trained with this gallery set having 124 individuals. After this, the trained CNN model weights are saved. We created two probe sets for model evaluation, one with clothing condition (coats) and the other with carrying condition (bags) of all 124 individuals. When we input these probe sets directly to our trained CNN model, then the results are poor. The CNN model fails to recognize the same 124 individuals on which it was trained because at this time, all 124 individuals come with an unknown walking condition. They wear jackets and carry bags; thus, these covariate factors alter the gait characteristics of all 124 people, causing the system to fail to detect them. As shown in Table 2, the accuracy of the trained CNN model is only 38% and 24.19% on probe set bags and coats. So as a solution, we recover the features set by employing cycle consistent generative adversarial networks (CCGANs). It converts the data of all 124 persons with bags and jackets to normal walk images and removes all covariate factors. At the time of training of CCGANs, the source and their corresponding target images are not

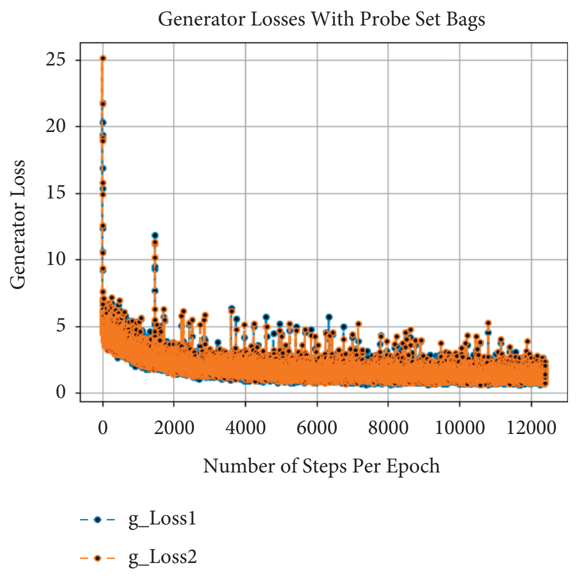


Figure 4: Loss of discriminators on number of steps per epoch with bags.

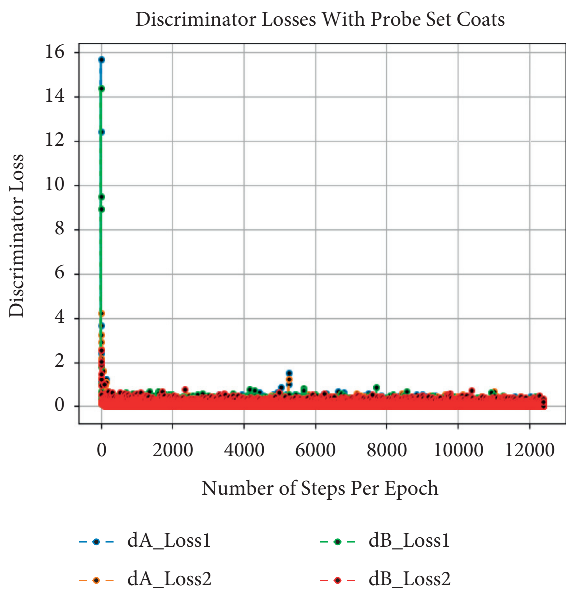


Figure 5: Loss of generators on number of steps per epoch with bags.

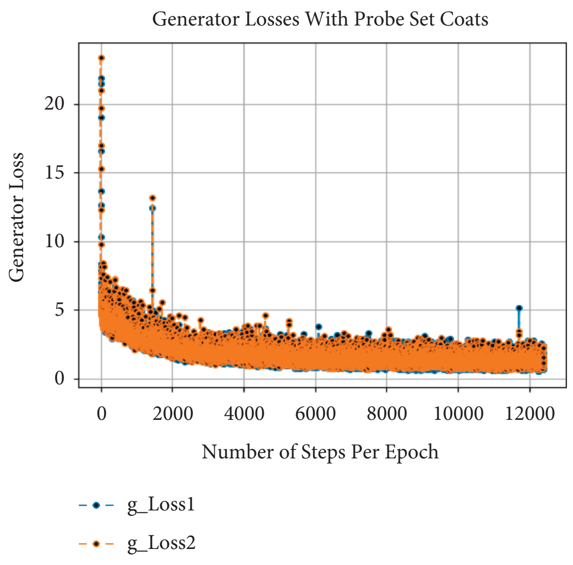


Figure 6: Loss of discriminators on number of steps per epoch with coats.

already known; that is, CCGANs automatically find the normal GEI for bag GEI of a particular individual with the help of Cycle Loss. CCGANs eliminate all covariate factors that make recognition difficult. Afterward, the translated data are fed into the CNN in order for the subject to be recognized. Table 2 shows the results of directly inputting covariate data to the CNN and the results of CNN + CCGANs. It is evident from the table that the accuracy of our proposed approach improves with CCGANs due to the reconstruction of gait features. On probe set with bags, we have achieved 79% accuracy, and with coats, 27.61% results are improved. Similarly, the accuracy, recall, and F1 score for bags are 0.72, 0.79, and 0.79, respectively, whereas on coats, they are 0.38, 0.51, and 0.41.

Besides this, at the time of training of CCGANs, loss values for generators and discriminators of both domains are shown in Figure 3. In Figure 3, the dA_loss1 is the loss of discriminator on real examples of domain A images, that is, bag or coat images, whereas dA_loss2 is the loss of discriminator of domain A on fake examples, i.e., artificially generated bags or coats images. Similarly, dB_loss1 and dB_loss2 are the losses of discriminators of domain B on real and fake examples, i.e., original and translated normal GEIs. Moreover, the loss of generators of both domains is also given as g_loss1 and g_loss2 in Figure 3. Each of these losses is a weighted average of cycle consistency loss L_{cyc} (both forward and backward) and adversarial losses L_{GAN} as given in equation (10). All these losses are recorded as per epochs. Furthermore, we have also calculated the loss per example

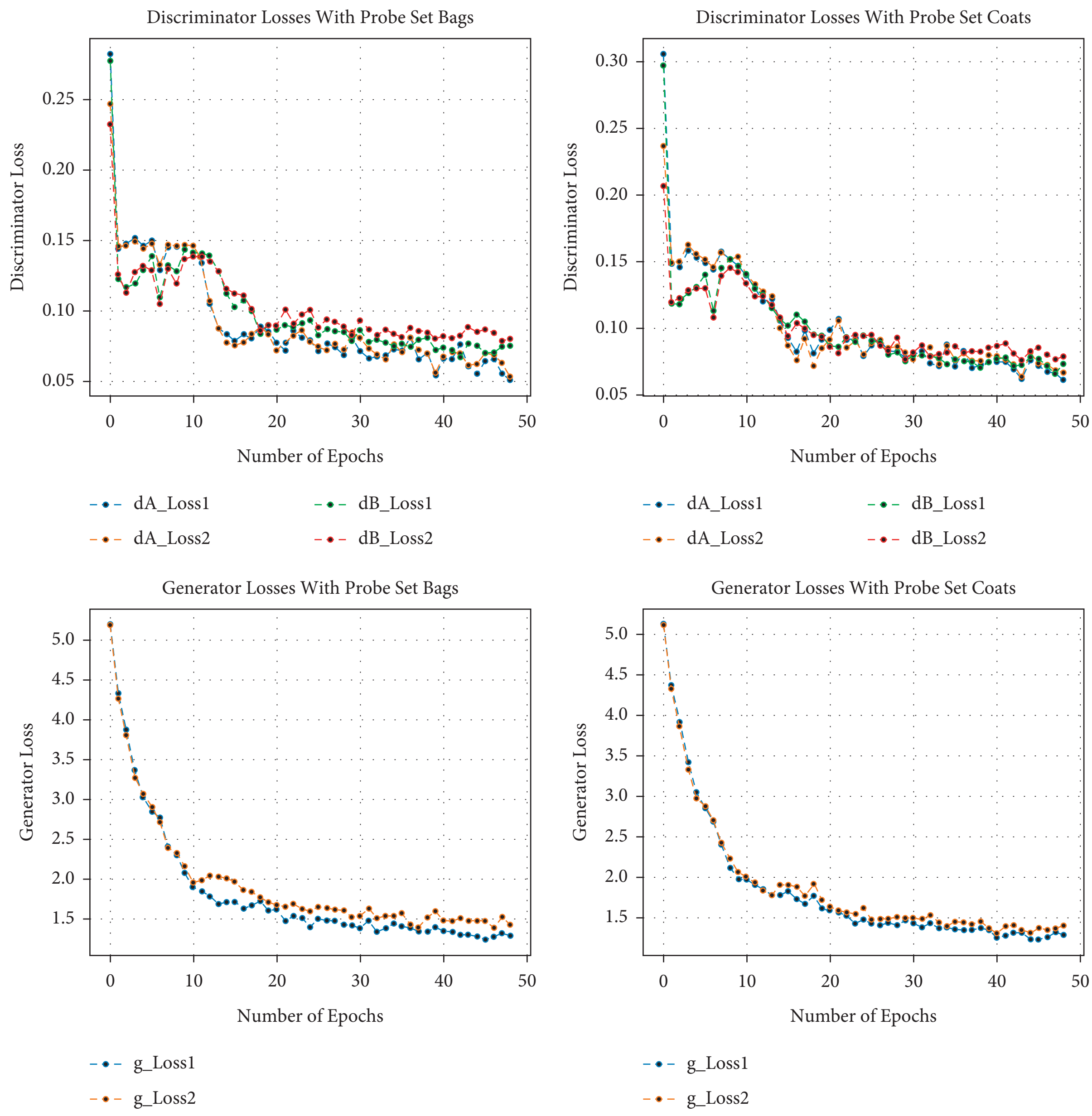


Figure 7: Loss of generators on number of steps per epoch with coats.

Table 3: Comparative analysis with the existing work under covariate conditions.

Sr. no.	Method	Normal	Bags	Coats (%)	Average accuracy (%)	Covariate
1	Bashir et al. [13]	100.0%	78.3%	44.4	74.2	Yes
2	Gupta et al. [18]	NA	86.2%	61.4	73.8	Yes
3	Hawas et al. [23]	97.6%	45.3%	49.6	64.1	Yes
4	Yu et al. [50]	95.97%	65.32%	42.74	68.01	Yes
5	Yao et al. [27]	NA	NA	38	38	Yes
6	Su et al. [25]	93.2%	72.8%	59.1	75.03	Yes
7	Proposed method	**97.98%**	**79%**	**51.8**	**76.26**	**Yes**

during training termed as the number of steps per epoch. The per example loss graphs for both of the probe sets is also given in Figures 4, 5, 6, and 7. In these figures, the x-axis represents the steps per epoch, and the y-axis denotes losses of discriminators and generators. We terminated training after 12400 steps (50 epochs) because there was no progress in losses beyond that. Table 3 also includes a comparison with prior approaches that take strict covariate conditions into account. From Table 3, it is observed that our proposed approach outperforms the existing approaches.

5. Conclusion

Gait Recognition without human involvement is a complex task in computer vision because all deep learning algorithms are not as much better to capture every possible dynamic walk environment. Generally, the computer vision algorithms are trained on a particular type of data to extract unique features and then conduct classification based on those extracted features. However, in the case of gait recognition without human involvement, the covariate factors change the gait features of individuals present in GEIs from those on which the deep learning model was trained. These covariate factors are unknown to the model, causing it to perform badly. So, this research work uses CCGANs to reconstruct the GEIs in which the same features are recovered on which model is trained. The CCGAN model is trained in an unsupervised manner to find a corresponding normal GEI for the bag/coat GEI of a particular individual. Then, the resultant translated images are tested by the CNN model. Moreover, we have achieved a very encouraging accuracy score of 79% in carrying conditions and also acceptable performance in clothing conditions. This work is further improved and extended using various sensors, to acquire gait data instead of only visual data and combine it with deep learning techniques to make the system more accurate and reliable.

Data Availability

The data sets generated during and analyzed during the current study are available from the corresponding author upon reasonable request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was funded by Institutional Fund Projects under grant no. IFPHI-033-611-2020. Therefore, authors gratefully acknowledge technical and financial support from the Ministry of Education and King Abdulaziz University, DSR, Jeddah, Saudi Arabia.

References

[1] J. L. Wayman, A. K. Jain, D. Maltoni, and D. Maio, *Biometric Systems: Technology, Design and Performance Evaluation*, Springer Science & Business Media, Berlin, Germany, 2005.

[2] C. Wan, L. Wang, and V. V. Phoha, "A survey on gait recognition," *ACM Computing Surveys (CSUR)*, vol. 51, pp. 1–35, 2018.

[3] Z. Zhang, M. Hu, and Y. Wang, "A survey of advances in biometric gait recognition," in *Proceedings of the Chinese Conference on Biometric Recognition*, pp. 150–158, Berlin, Germany, 2011.

[4] M. Bukhari, K. B. Bajwa, S. Gillani et al., "An efficient gait recognition method for known and unknown covariate conditions," *IEEE Access*, vol. 9, pp. 6465–6477, 2020.

[5] 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, Venice, Italy, October 2017.

[6] H. Ali, J. Dargham, C. Ali, and E. G. Moung, "Gait recognition using gait energy image," *International Journal of Signal Processing, Image Processing and Pattern Recognition*, vol. 4, pp. 141–152, 2011.

[7] I. Goodfellow, J. Pouget-Abadie, M. Mirza et al., "Generative adversarial nets," *Advances in Neural Information Processing Systems*, vol. 27, 2014.

[8] 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. 1125–1134, Venice, Italy, October 2017.

[9] C. Yam, M. S. Nixon, and J. N. Carter, "Automated person recognition by walking and running via model-based approaches," *Pattern Recognition*, vol. 37, no. 5, pp. 1057–1072, 2004.

[10] F. Tafazzoli and R. Safabakhsh, "Model-based human gait recognition using leg and arm movements," *Engineering Applications of Artificial Intelligence*, vol. 23, no. 8, pp. 1237–1246, 2010.

[11] A. F. Bobick and A. Y. Johnson, "Gait recognition using static, activity-specific parameters," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR*, pp. 8–14, Kauai, HI, USA, December 2001.

[12] R. Anusha and C. D. Jaidhar, "Clothing invariant human gait recognition using modified local optimal oriented pattern binary descriptor," *Multimedia Tools and Applications*, vol. 79, no. 3-4, pp. 2873–2896, 2020.

[13] K. Bashir, T. Xiang, and S. Gong, "Gait recognition without subject cooperation," *Pattern Recognition Letters*, vol. 31, no. 13, pp. 2052–2060, 2010.

[14] M. Alotaibi and A. Mahmood, "Improved gait recognition based on specialized deep convolutional neural network," *Computer Vision and Image Understanding*, vol. 164, pp. 103–110, 2017.

[15] J. Han and B. Bhanu, "Individual recognition using gait energy image," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, pp. 316–322, 2005.

[16] A. O. Lishani, L. Boubchir, E. Khalifa, and A. Bouridane, "Human gait recognition using GEI-based local multi-scale feature descriptors," *Multimedia Tools and Applications*, vol. 78, no. 5, pp. 5715–5730, 2019.

[17] M. Rokanujjaman, M. S. Islam, M. A. Hossain, M. R. Islam, Y. Makihara, and Y. Yagi, "Effective part-based gait identification using frequency-domain gait entropy features," *Multimedia Tools and Applications*, vol. 74, no. 9, pp. 3099–3120, 2015.

[18] S. K. Gupta, G. M. Sultaniya, and P. Chattopadhyay, "An efficient descriptor for gait recognition using spatio-temporal cues," in *Emerging Technology in Modelling and Graphics*, pp. 85–97, Springer, Berlin, Germany, 2020.

[19] Z. Ali, A. Irtaza, and M. Maqsood, "An efficient U-Net framework for lung nodule detection using densely connected dilated convolutions," *The Journal of Supercomputing*, pp. 1–22, 2021.

[20] M. Maqsood, S. Yasmin, I. Mehmood, M. Bukhari, and M. Kim, "An efficient DA-Net architecture for lung nodule segmentation," *Mathematics*, vol. 9, no. 13, p. 1457, 2021.

[21] M. Maqsood, M. Bukhari, Z. Ali et al., "A residual-learning-based multi-scale parallel-convolutions-assisted efficient

CAD system for liver tumor detection," *Mathematics*, vol. 9, no. 10, p. 1133, 2021.

[22] A. G. Salman, B. Kanigoro, and Y. Heryadi, "Weather forecasting using deep learning techniques," in *Proceedings of the 2015 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, pp. 281–285, Depok, Indonesia, October 2015.

[23] A. R. Hawas, H. A. El-Khobby, M. Abd-Elnaby, and F. E. Abd El-Samie, "Gait identification by convolutional neural networks and optical flow," *Multimedia Tools and Applications*, vol. 78, no. 18, pp. 25873–25888, 2019.

[24] G. M. Linda, G. Themozhi, and S. R. Bandi, "Color-mapped contour gait image for cross-view gait recognition using deep convolutional neural network," *International Journal of Wavelets, Multiresolution and Information Processing*, vol. 18, no. 01, Article ID 1941012, 2020.

[25] J. Su, Y. Zhao, and X. Li, "Deep metric learning based on center-ranked loss for gait recognition," in *Proceedings of the ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4077–4081, Barcelona, Spain, May 2020.

[26] G. Huang, Z. Lu, C. M. Pun, and L. Cheng, "Flexible gait recognition based on flow regulation of local features between key frames," *IEEE Access*, vol. 8, pp. 75381–75392, 2020.

[27] L. Yao, W. Kusakunniran, Q. Wu, J. Zhang, Z. Tang, and W. Yang, "Robust gait recognition using hybrid descriptors based on skeleton gait energy image," *Pattern Recognition Letters*, vol. 150, 2019.

[28] H. Ling, J. Wu, P. Li, and J. Shen, "Attention-aware network with latent semantic analysis for clothing invariant gait recognition," *Computers, Materials & Continua*, vol. 60, no. 3, pp. 1041–1054, 2019.

[29] X. Wu, T. Yang, and Z. Xia, "Gait recognition based on densenet transfer learning," *International Journal of Science and Environment*, vol. 9, pp. 1–14, 2020.

[30] X. Shu, L. Zhang, Y. Sun, and J. Tang, "Host–parasite: graph LSTM-in-LSTM for group activity recognition," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, pp. 663–674, 2020.

[31] J. Tang, X. Shu, R. Yan, and L. Zhang, "Coherence constrained graph LSTM for group activity recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[32] M. H. Kabir, K. Thapa, J. Y. Yang, and S. H. Yang, "State-space based linear modeling for human activity recognition in smart space," *Intelligent Automation and Soft Computing*, vol. 25, pp. 673–681, 2019.

[33] X. Shu, L. Zhang, G. J. Qi, W. Liu, and J. Tang, "Spatiotemporal co-attention recurrent neural networks for human-skeleton motion prediction," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[34] S. T. Choe, W. D. Cho, J. H. Kim, and K. H. Kim, "Reducing operational time complexity of k-NN algorithms thin; clustering in wrist-activity recognition," *Intelligent Automation & Soft Computing*, vol. 26, no. 4, pp. 679–691, 2020.

[35] W. Song, J. Yu, X. Zhao, and A. Wang, "Research on action recognition and content analysis in videos based on DNN and MLN," *Computers, Materials & Continua*, vol. 61, no. 3, pp. 1189–1204, 2019.

[36] J. Jo, H. G. Kim, I. C. Park, B. C. Jung, and H. Yoo, "Modified viterbi scoring for HMM-based speech recognition," *Intelligent Automation and Soft Computing*, vol. 25, pp. 351–358, 2019.

[37] C. H. Chen, C. Y. Chen, and N. Y. Liu, "Hardware design of codebook-based moving object detecting method for dynamic gesture recognition," *Intelligent Automation and Soft Computing*, vol. 25, pp. 375–384, 2019.

[38] J. Niu, Y. Jiang, Y. Fu, T. Zhang, and N. Masini, "Image deblurring of video surveillance system in rainy environment," *Computers, Materials & Continua*, vol. 65, no. 1, pp. 807–816, 2020.

[39] N. Wang, M. He, J. Sun et al., "IA-PNCC: noise processing method for underwater target recognition convolutional neural network," *Computers, Materials & Continua*, vol. 58, no. 1, pp. 169–181, 2019.

[40] V. Kearney, B. P. Ziemer, A. Perry et al., "Attention-aware discrimination for MR-to-CT image translation using cycle-consistent generative adversarial networks," *Radiology: Artificial Intelligence*, vol. 2, no. 2, Article ID e190027, 2020.

[41] K. Armanious, C. Jiang, S. Abdulatif, T. Küstner, S. Gatidis, and B. Yang, "Unsupervised medical image translation using cycle-MedGAN," in *Proceedings of the 2019 27th European Signal Processing Conference (EUSIPCO)*, pp. 1–5, A Coruna, Spain, September 2019.

[42] S. Yu, H. Chen, E. B. Garcia Reyes, and N. Poh, "Gaitgan: invariant gait feature extraction using generative adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 30–37, Honolulu, HI, USA, July 2017.

[43] Y. He, J. Zhang, H. Shan, and L. Wang, "Multi-task GANs for view-specific feature learning in gait recognition," *IEEE Transactions on Information Forensics and Security*, vol. 14, pp. 102–113, 2018.

[44] X. Li, Y. Makihara, C. Xu, Y. Yagi, and M. Ren, "Gait recognition invariant to carried objects using alpha blending generative adversarial networks," *Pattern Recognition*, vol. 105, Article ID 107376, 2020.

[45] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[46] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, "Adversarial autoencoders," 2015, https://arxiv.org/abs/1511.05644.

[47] S. Yu, D. Tan, and T. Tan, "A framework for evaluating the effect of view angle, clothing and carrying condition on gait recognition," in *Proceedings of the 18th International Conference on Pattern Recognition (ICPR'06)*, pp. 441–444, Hong Kong, China, August 2006.

[48] A. Kalsoom, M. Maqsood, M. A. Ghazanfar, F. Aadil, and S. Rho, "A dimensionality reduction-based efficient software fault prediction using fisher linear discriminant analysis (FLDA)," *The Journal of Supercomputing*, vol. 74, no. 9, pp. 4568–4602, 2018.

[49] R. Anusha and C. D. Jaidhar, "Human gait recognition based on histogram of oriented gradients and haralick texture descriptor," *Multimedia Tools and Applications*, vol. 79, no. 11-12, pp. 8213–8234, 2020.

[50] S. Yu, H. Chen, Q. Wang, L. Shen, and Y. Huang, "Invariant feature extraction for gait recognition using only one uniform model," *Neurocomputing*, vol. 239, pp. 81–93, 2017.

Hindawi
Computational Intelligence and Neuroscience
Volume 2021, Article ID 7018035, 16 pages
https://doi.org/10.1155/2021/7018035

Research Article

NSD-SSD: A Novel Real-Time Ship Detector Based on Convolutional Neural Network in Surveillance Video

Jiuwu Sun,[1] **Zhijing Xu**,[1] **and Shanshan Liang**[2]

[1]*College of Information Engineering, Shanghai Maritime University, Shanghai 201306, China*
[2]*School of Physics and Electronics, Shandong Normal University, Jinan 250358, China*

Correspondence should be addressed to Zhijing Xu; zjxu@shmtu.edu.cn

Received 27 May 2021; Revised 13 August 2021; Accepted 18 August 2021; Published 14 September 2021

Academic Editor: Francisco Gomez-Donoso

With the rapid development of the marine industry, intelligent ship detection plays a very important role in the marine traffic safety and the port management. Current detection methods mainly focus on synthetic aperture radar (SAR) images, which is of great significance to the field of ship detection. However, these methods sometimes cannot meet the real-time requirement. To solve the problems, a novel ship detection network based on SSD (Single Shot Detector), named NSD-SSD, is proposed in this paper. Nowadays, the surveillance system is widely used in the indoor and outdoor environment, and its combination with deep learning greatly promotes the development of intelligent object detection and recognition. The NSD-SSD uses visual images captured by surveillance cameras to achieve real-time detection and further improves detection performance. First, dilated convolution and multiscale feature fusion are combined to improve the small objects' performance and detection accuracy. Second, an improved prediction module is introduced to enhance deeper feature extraction ability of the model, and the mean Average Precision (mAP) and recall are significant improved. Finally, the prior boxes are reconstructed by using the *K*-means clustering algorithm, the Intersection-over-Union (IoU) is higher, and the visual effect is better. The experimental results based on ship images show that the mAP and recall can reach 89.3% and 93.6%, respectively, which outperforms the representative model (Faster R-CNN, SSD, and YOLOv3). Moreover, our model's FPS is 45, which can meet real-time detection acquirement well. Hence, the proposed method has the better overall performance and achieves higher detection efficiency and better robustness.

1. Introduction

With the rapid development of the shipping industry, there are more frequent human activities on the ocean in recent years. Therefore, robust ship detection is strongly needed to meet the demand. Currently, ship detection is used in port transportation management, sea area monitoring over illegal activities, and ship abnormal behavior detection for navigation safety. Modern radar target tracking equipment and ship automatic identification systems are mainly based on positioning, and thus, ship detection needs substantial improvements. In response to these problems, many researchers have used traditional machine learning methods to explore this field in search of better results. For example, they used features of ships combined with classifiers [1, 2]. Although these methods achieve good results, they require manual extraction of features and a classifier with good performance, which needs further validation in terms of efficiency and accuracy. Fortunately, the development of deep learning has enabled object detection to be widely used in many scenarios, such as surveillance security and autonomous driving. In 2019, Jiao et al. [3] provided a comprehensive analysis of the current state and future trends of deep learning-based object detection. Convolutional Neural Networks (CNN) can effectively learn the corresponding features from massive samples, which avoids the complicated feature extraction process and achieves higher accuracy. In 1998, Lecun et al. [4] proposed LeNet-5 and achieved success in the recognition of handwritten characters. Since then, the performance of CNNs has been improved with the appearance of deeper and more complex CNNs such as AlexNet [5], VGGNet [6], GoogLeNet [7], ResNet [8], and DenseNet [9]. In 2020, Abdollahi et al. [10] used a generative adversarial network (GAN) architecture to extract building footprints from high-resolution aerial images. However, the algorithms of regular CNNs combined with feature pyramid networks (FPN) have become a new focus in the field of object detection. The object detection algorithms currently mainly include two technical routes: two-stage detection and one-stage detection. The two-stage detection is divided into two steps. First obtain the region proposals, and then, these region proposals are classified and regressed to get the final detection results. Two-stage detectors mainly include R-CNN [11], SPP-Net [12], Fast R-CNN [13], Faster R-CNN [14], and Mask R-CNN [15]. For one-stage detection, it treats the object detection problem as a regression problem. A unified CNN completes the object classification and location, which is an end-to-end target detection solution. One-stage detectors mainly include OverFeat [16], SSD [17], and YOLO [18–21]. Many scholars proposed improved YOLOv3 and SSD for object detection and obtained outstanding detection performance [22, 23]. The two-stage detection algorithm such as Faster R-CNN has high accuracy, but its region proposal network (RPN) is time-consuming and therefore reduces the detection efficiency. On the contrary, although the YOLO series has a great advantage in terms of detection speed, they cannot achieve high accuracy.

The SSD is used as a one-stage detector and introduces a multiscale feature layer for object detection, which has faster detection speed but accuracy needs to be improved. In this paper, the SSD is applied to ship detection and several improvements are used to improve the overall performance of the network.

(1) To address the problem of poor performance of small target detection, we apply a dilated convolution on the low-level feature layer to expand the receptive field so that the low-level feature layer can also contain more feature information. At the same time, we perform multiscale fusion on the original feature layers after up-sampling so that the network can make full use of the contextual information. (2) We introduce a residual structure in the prediction module of the network to enable the network to extract deeper dimensional feature information for better classification and regression. (3) We use the K-means clustering algorithm to reconstruct the prior bounding box so as to obtain a more suitable scale and aspect ratio, which can improve both the visual effect and the efficiency of ship detection. Finally, we propose a new SSD-based network, called NSD-SSD, which is significantly better than the original SSD. Compared with SSD and other detection networks, the proposed network provides a good trade-off between real-time detection and accuracy.

The rest of this paper is organized as follows. In Section 2, we introduce the related work of the ship detection. In Section 3, we give detailed program of our proposed approach. Section 4 outlines the experimental results and comparisons against other state-of-the-art methods. Finally, conclusions are made in Section 5.

2. Related Work

This paper categorizes the previous work of ship object detection to traditional methods and deep learning methods.

The traditional detection methods include two types. (1) Ship-radiated noise-based methods: Kang et al. [24] proposed a multiple classifier fusion algorithm based on many-person decision theory to identify ship radiated noise, with accuracy rate of over 96%. Zhao et al. [25] proposed a decision tree support vector machine (SVM) classification method based on the ship-radiated noise multidimension feature vector for the measured radiated noise of three kinds of ship targets. Luo and Wang [26] used the time-frequency range characteristics of ship noise to distinguish ship's stern, ship's mid-aft, and ship's middle part to complete the positioning and identification of ship targets. Peng et al. [27] proposed a ship-radiated noise model based on the winger's higher-order spectrum for feature extraction. (2) Ship structure and shape characteristics-based methods: Zhu et al. [28] proposed a novel hierarchical method of ship detection from spaceborne optical image based on shape and texture features, and this method can effectively distinguish ships from nonships on the optical image dataset. Liu et al. [29] used segmentation and shape analysis to detect inshore ships and proved their method was effective and robust under various situations. Shi et al. [30] proposed an approach involving a predetection stage and an accurate detection stage to detect ships in a coarse-to-fine manner in high-resolution optical images. Wang et al. [31] proposed a detection method based on DoG (Difference of Gaussian) preprocessing and shape features to detect ship targets in remote sensing images.

Most of the traditional methods use manually extracted features, which will lead to low efficiency and high time consumption. At the same time, even if a classifier with good performance is used to classify these features, the accuracy cannot meet the actual demand. Therefore, the recognition rate of these methods in complex environmental background and multivessel classification is not ideal.

The deep learning detection methods: with the boom development of deep learning, many ship object detection methods based on deep CNN have been proposed. Zou et al. [32] proposed an improved SSD algorithm based on MobilenetV2 [33] and finally achieved better detection results in three types of ship images. Zhao et al. [34] proposed a new network architecture based on the Faster R-CNN by using squeeze and excitation for ship detection in SAR images. Shao et al. [35] proposed a saliency-aware CNN framework and coastline segmentation method to improve the accuracy and robustness of ship detection under complex seashore surveillance conditions. Nie et al. [36] proposed an improved Mask R-CNN model, which can accurately detect and segment ships from remote sensing images at the pixel level. Guo et al. [37] proposed a novel SSD network structure to improve the semantic information by deconvoluting high-level features into a low-level feature and then fusing it with original low-level features, and the model performed well on both the PASCAL VOC and railway datasets. Huang et al. [38] proposed a new network by referring to the feature extraction layer of YOLOv2 and feature pyramid network of YOLOv3, and the new network model can detect seven types of ships. Zhao et al. [39] proposed the Attention Receptive Pyramid Network (ARPN), which detected multiscale ships in SAR images. Li et al. [40] proposed a new method, combining the Saliency Estimation Algorithms (SEAs) and the Deep CNN (DCNN) object detection to ensure the extraction of large-scale ships. In 2021, Zhao et al. [41] proposed a feature pyramid enhancement strategy (FPES) and a cascade detection mechanism to improve SSD, and the improved model can be applied to vehicle detection quickly and efficiently.

In short, although the existing ship target detection methods have made major breakthroughs, they still have certain limitations. Firstly, the low-level feature map contains less semantic information but can accurately present the location of the target. In contrast, high-level feature maps contain rich semantic information but cannot accurately display the location of objects. In addition, the previous methods cannot extract the features of small objects well. In this paper, we use a multiscale feature fusion algorithm, which considers the ability of the entire network to combine the context information and improve small target detection performance. In addition, we have also improved the prediction module and the settings of prior boxes. Finally, we test the improved model on the ship dataset.

3. Materials and Methods

3.1. Single-Shot Multibox Detector. Figure 1 shows the SSD network structure diagram with a backbone network VGG-16. VGG-16 has stable network structure and good feature extraction capabilities. The SSD network converts FC6 and FC7 in VGG-16 into convolutional layers, removes all Dropout layers and FC8 layers, and adds four additional convolutional layers: Conv6, Conv7, Conv8, and Conv9. The feature pyramid structure is to detect objects of different sizes. In the process of detection, a large number of prior boxes are usually generated, and these prior boxes have multiple predefines scales and ratios. Finally, it is required to apply a Nonmaximum Suppression (NMS) process to obtain the final test results. The biggest advantage of the SSD network is that classification and regression are carried out at the same time, which improves the detection speed compared with other models such as Faster R-CNN.

3.2. Our Proposed Network. The overall architecture of the Novel Ship Detection SSD (NSD-SSD) is shown in Figure 2. From the figure, the architecture mainly is formed by three parts, a dilated convolution layer, a multiscale feature fusion layer, and a prediction layer. In addition, the prior boxes are reconstructed within this network. Ship images are sent to the NSD-SSD network for a series of operations, and finally, the specific location and type of ship can be obtained.

To understand the features extracted by the network more clearly, a visualization of the feature maps is given in Figure 3. In the figure, from left to right, the input image, the feature maps extracted by SSD, and the feature maps extracted after feature layer fusion are shown. From the figure, we can see that the feature maps extracted by the original SSD network lack rich semantic information. For example, the main characteristics of the low-level feature layer Conv4_3 are small perceptual field and too poor ability to extract target features. However, after the dilated convolution and features fusion, the feature information of the target is greatly enriched. Similarly, all other scale layers also extract a large amount of meaningful contextual information after feature fusion, which greatly improves the accuracy of object detection.

3.2.1. Dilated Convolution Layer. Traditional SSD network mainly uses low-level feature layer Conv4_3 to detect small objects. However, due to insufficient feature extraction in the Conv4_3 layer, the detection effect of small objects is not ideal. To address this issue, we use dilated convolution to map high-dimensional features to low-dimensional input. In this paper, we choose the lower-level feature layer Conv3_1 for dilated convolution and merge it with Conv4_3 for feature fusion. In this way, the range of the receptive field can be enlarged without loss of image detail information and obtains more global information.

Dilated convolution is to inject dilation on map of the standard convolution to increase the receptive field. The dilated convolution has another hyperparameter called the dilation rate, which refers to the number of intervals of the convolution kernel. Assuming that the original convolution kernel is f and the dilation rate is α, the new convolution kernel size n after dilated convolution is

$$n = \alpha \times (f - 1) + 1. \quad (1)$$

The receptive field size r after dilated convolution is

$$r = \left[2^{(\alpha/2)+2} - 1\right] \times \left[2^{(\alpha/2)+2} - 1\right]. \quad (2)$$

Suppose that there is a dilated convolution with $f = 3$ and $\alpha = 1$, which is equivalent to a standard convolution. Its receptive field is 3×3. When $f = 3$ and $\alpha = 2$, according to equations (1) and (2), its new convolution kernel is 5×5, and the receptive field size is expanded to 7×7 without losing detailed information.

In this paper, we choose the Conv3_1 layer for dilated convolution. The original kernel is 3×3, stride is 2, pad is 2, and dilation rate is 2. From equation (1), the new convolution kernel is 5×5. The original feature map of Conv3_1 layer is $75 \times 75 \times 256$. After performing dilated convolution, it obtains a feature map size which is $38 \times 38 \times 512$. From equation (2), the receptive filed is 7×7. The Conv3_1 layer undergoes feature map fusion with the Conv4_3 layer after dilated convolution. There are two main ways of feature map fusion: additive fusion and cascade fusion. Because the cascade fusion has a small amount of calculation and high accuracy, in this paper, we choose cascade fusion method. Figure 4 shows the process of feature map fusion.

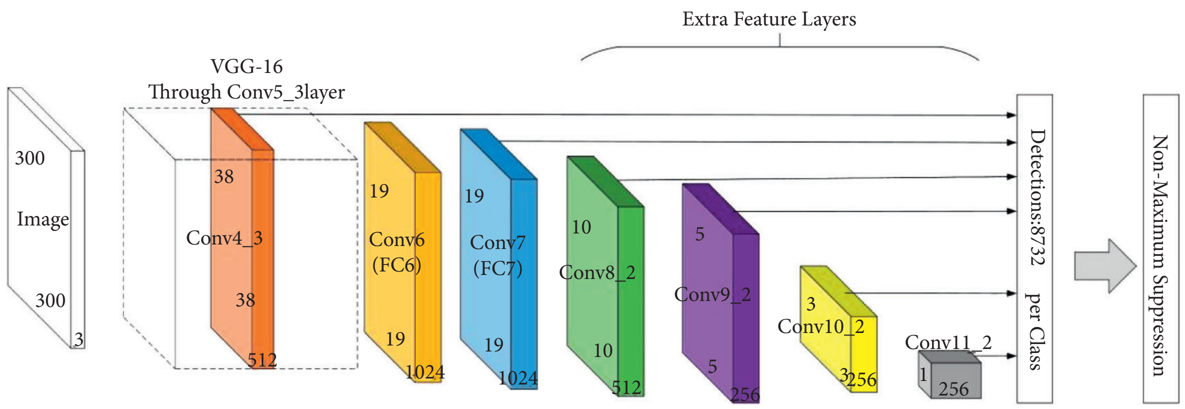


Figure 1: SSD network structure diagram.

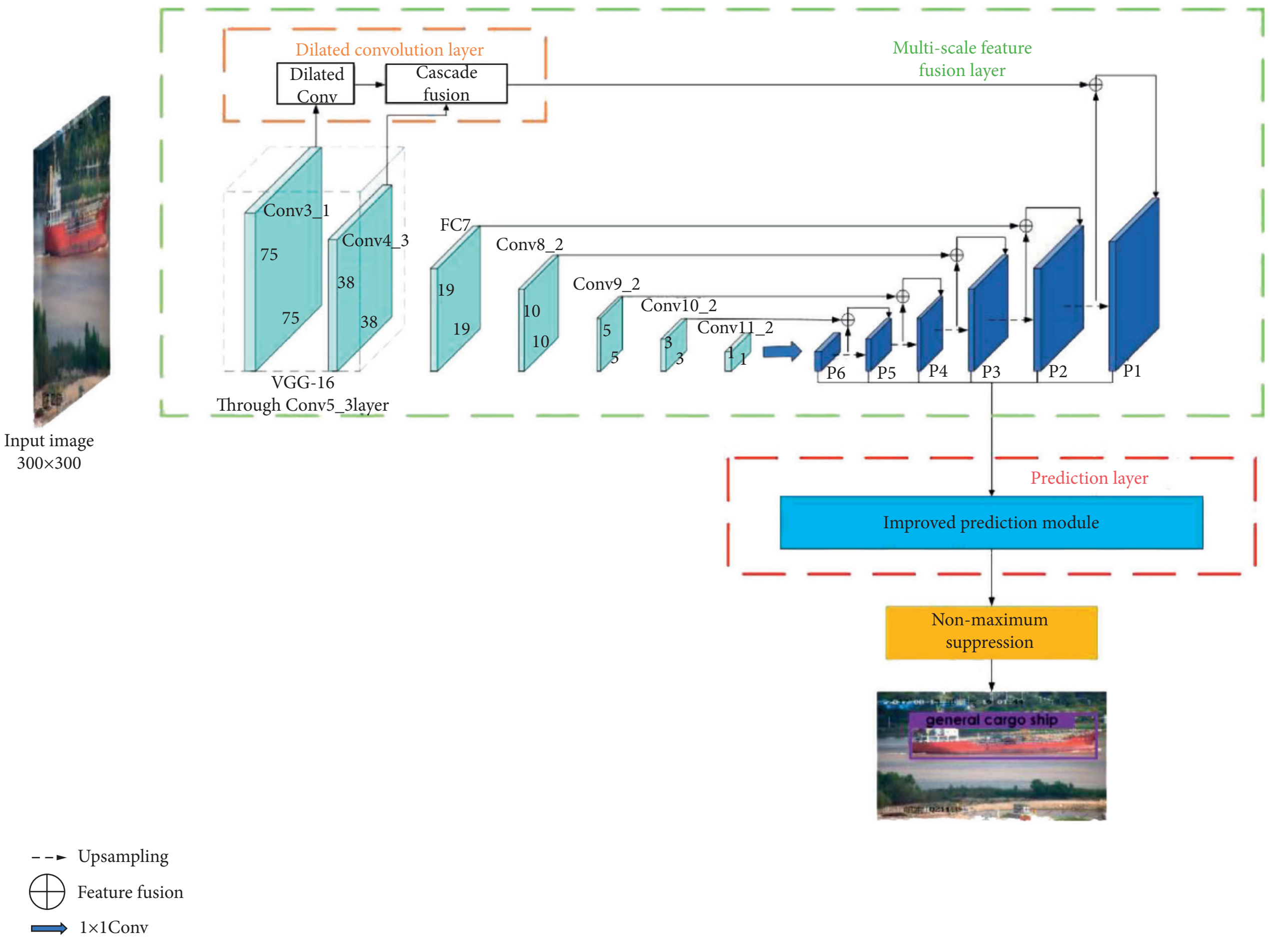


Figure 2: The architecture of the proposed novel ship detection SSD (NSD-SSD).

To better explain how the dilated convolution improves the performance of the network with the addition of feature maps, Figure 5 shows the feature maps of the image before and after the dilated convolution.

In the figure, (a) is the original image, (b) are the feature maps of Conv4_3 in the SSD network, and (c) are the feature maps with dilated convolution and feature fusion. The original features of the Conv4_3 activation area and perceptual field are small and cannot detect the ship targets at the corresponding scales well. The original features of the Conv4_3 activation area and perceptual field are small and cannot detect the ship targets at the corresponding scales well. On the contrary, the dilated convolution and feature fusion are able to more richly extract the texture and detail features on the low-level feature maps, and the contours and shapes are more clearly distinguished.

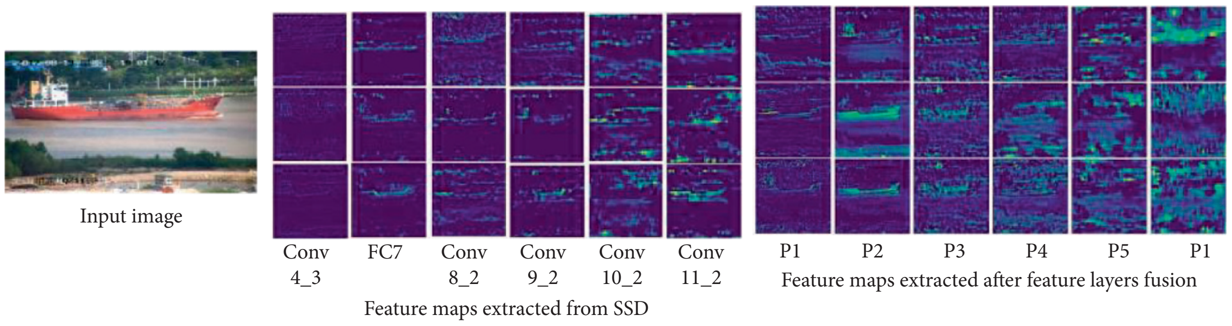

Input image
Conv 4_3
FC7
Conv 8_2
Conv 9_2
Conv 10_2
Conv 11_2
Feature maps extracted from SSD
P1
P2
P3
P4
P5
P1
Feature maps extracted after feature layers fusion

FIGURE 3: Feature maps of ship images extracted by the network.

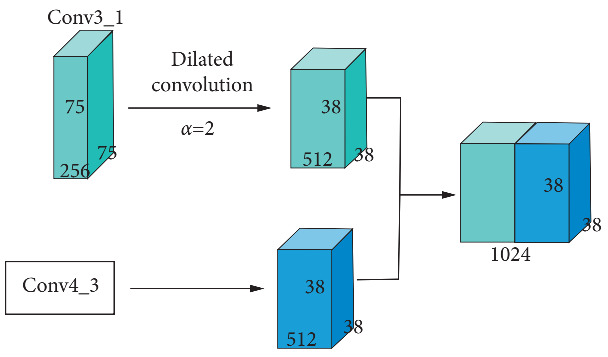


FIGURE 4: Fusion process of the Conv3_1 layer with the Conv4_3 layer after dilated convolution.

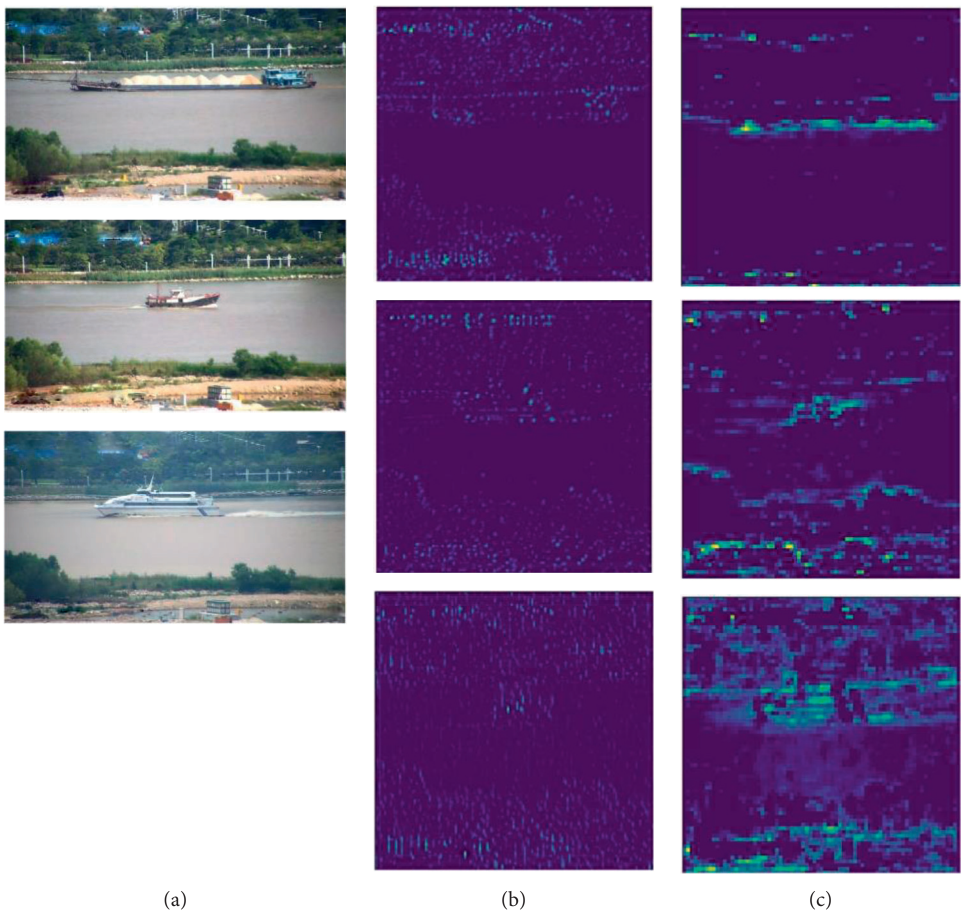


FIGURE 5: Comparison of the feature maps in the original SSD and after the dilated convolution.

3.2.2. Multiscale Feature Fusion Layer. The original SSD network uses the Feature Pyramid Network (FPN) to detect different feature layers so that it can adapt to different object sizes. Although this detection method provides the possibility of multiscale object detection, it does not consider the combination of shallow features and deep features. In this study, on the basis of the original SSD network, we introduce a multiscale feature fusion mechanism. This method can synthesize shallow high-resolution features and deep semantic features to make joint decisions. The green dotted box in Figure 2 shows the specific fusion connections of different feature layers. The left half of the figure is the original SSD network feature layer, and the right half is the fused feature layer. The specific implementation process of this feature fusion method will be described in detail below. First, perform 1×1 convolution of Conv11_2 to obtain P6, then perform up-sampling of P6, and finally perform 1×1 convolution of Conv10_2 with the feature layer obtained by up-sampling P6 to obtain P5. The purpose of up-sampling here is to obtain the feature map of the size required for fusion. After the same fusion process, the fused feature layers are successively P4, P3, P2, and P1. In this way, the combination of shallow features and deep features is considered comprehensively, and it is possible to improve the detection accuracy. P1 is formed by fusion of dilated convolutional layer and P2 up-sampling. The parameters of the prediction layer are shown in Table 1.

3.2.3. Improved Prediction Module. The SSD network uses a set of convolution filters at each effective feature layer to obtain prediction results. For each effective feature layer with a size of $h \times w$ with $d$channels, use a 3×3 convolution operation on each route to obtain the score of each category and the change of each prior bounding box.

MS-CNN [42] points out that improving the subnetwork of each task can improve the accuracy. DSSD [43] follows this principle and proposes an improved prediction module, and experimental results show that this method can improve detection accuracy. Therefore, we transplant the idea of DSSD into our network model to better improve the detection performance. The prediction layer corresponds to the red box in Figure 2. That is, on the basis of SSD, the original structure is changed to a residual module. The residual prediction block allows the use of 1×1 convolution to predict the score of each category and the changes of prior boxes. The structure of the original predictor and the improved predictor are shown in Figure 6. In this way, deeper dimensional features can be extracted for classification and regression.

3.2.4. Reconstruction of Regional Prior Box. The performance of deep learning object detection algorithms largely depends on the quality of feature learning driven by training data. In the SSD object detection task, the training data is the regional prior box. The SSD network has selected a total of 6 effective feature layers as the prediction layer, the sizes of which are (38, 38), (19, 19), (10, 10), (5, 5), (3, 3), and (1, 1), but the number of a prior bounding boxes set on each feature map is different. The prior bounding box has two hyperparameters: scale and aspect ratio. The scale of a prior bounding box in each prediction layer is

$$S_k = S_{\min} + \frac{S_{\max} - S_{\min}}{m-1}(k-1), \quad k \in [1, m]. \tag{3}$$

Among them, m refers to the number of feature maps ($m = 6$ in the SSD algorithm), s_k represents the ratio of the prior box size of the kth feature map to the picture, $S_{\min}$ represents the minimum value of the ratio, and the value is 0.2, and $S_{\max}$ indicates the maximum value of the ratio, and the value is 0.9. The aspect ratio of the prior bounding box is generally set to $a_r = \{1, 2, 3, 1/2, 1/3\}$. The width and height of the prior bounding box are as follows:

$$\begin{cases} w_k^a = S_k \sqrt{a_r}, \\ h_k^a = \dfrac{S_k}{\sqrt{a_r}}. \end{cases} \tag{4}$$

By default, each feature map will have a prior bounding box with $a_r = 1$ and a scale of S_k. In addition, the prior bounding box with a scale of $S_k' = \sqrt{S_k S_{k+1}}$ will be added. In this way, each feature map has two square prior bounding boxes with an aspect ratio of 1 but different sizes. The maximum side length of the square prior bounding box is $S_k' = \sqrt{S_k S_{k+1}}$, and the minimum side length is S_k. Table 2 lists the min-size and max-size of the prior bounding boxes used in this paper.

As shown in Figure 7, 4 prior bounding boxes are generated, two squares (red dashed line) and two rectangles (blue dashed line). At this time, the aspect ratio $a_r = \{1, 2\}$. Among them, $S_k * 300$ is the side length of the small square and $\sqrt{S_k S_{k+1}} * 300$ is the side length of the large square. 300 is the size of the input image in the SSD algorithm. The width and height of the corresponding two rectangles are

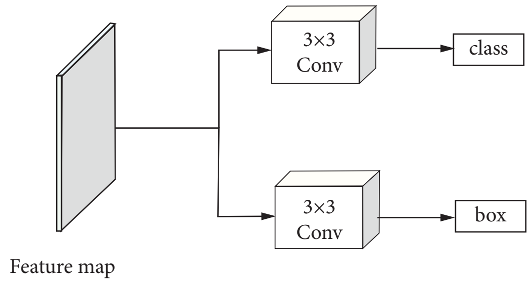
$$\begin{cases} \sqrt{a_r} * S_k * 300, & \dfrac{1}{\sqrt{a_r}} * S_k * 300, \\ \sqrt{\dfrac{1}{a_r}} * S_k * 300, & \dfrac{1}{\sqrt{1/a_r}} * S_k * 300. \end{cases} \tag{5}$$

When 6 prior bounding boxes are to be generated, the aspect ratio $a_r = \{1, 2, 3\}$. The center point of each prior box is $(i + 0.5/|f_k|, j + 0.5/|f_k|)$, i and $j \in [0, |f_k|]$, and f_k is the size length of the feature map. In this paper, $f_k = \{38, 19, 10, 5, 3, 1\}$. Table 3 shows the detailed parameters of the prior bounding boxes of the SSD algorithm.

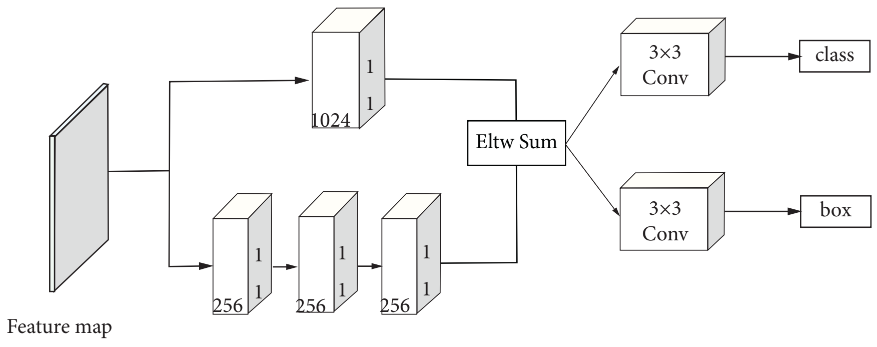
In the SSD algorithm, the scale and aspect ratio of the prior boxes in the network cannot be obtained through learning, but manually set. Since each feature map in the network uses different prior bounding boxes in scale and shape, the debugging process is very dependent on experience. In this paper, we use the K-means algorithm to predict the scale and proportion of the prior bounding box to improve the detection efficiency of the network. The standard K-means clustering algorithm uses Euclidean distance to measure distance. But if Euclidean distance is

Table 1: Parameters of the prediction layer.

Prediction layer	Kernel size	Padding	Kernel numbers	Strides	Feature map
P1	3×3	1	1024	1	38×38
P2	3×3	1	1024	1	19×19
P3	3×3	1	512	1	10×10
P4	3×3	1	256	1	5×5
P5	3×3	1	256	1	3×3
P6	3×3	1	256	1	1×1



(a)



(b)

Figure 6: The prediction process of the feature layer. (a) The original SSD predictor: obtain the score of each category and the change of the prior box after two convolution routes. (b) The improved predictor: add the residual structure on the basis of (a) to obtain the prediction result.

Table 2: Size of prior bounding boxes for different feature layers.

Feature layer	Min-size	Max-size
Conv4_3	30	60
FC7	60	111
Conv8_2	111	162
Conv9_2	162	213
Conv10_2	213	264
Conv11_2	264	315

used here, the larger boxes will produce more errors than the small boxes. Therefore, we use other distance measurement methods, and the specific equation is as follows:

$$\begin{aligned} d(\text{box}, \text{centroid}) &= 1 - \text{IOU}(\text{box}, \text{centroid}) \\ &= 1 - \text{IOU}\left[\left(x_j, y_j, w_j, h_j\right), \left(x_j, y_j, W_i, H_i\right)\right]. \end{aligned} \quad (6)$$

IoU is the intersection ratio between the regional prior bounding boxes and the ground truth boxes, and we expect a larger IoU. The purpose of clustering is that the prior bounding boxes and the adjacent ground truth have a large IoU value. Equation (6) just ensures that the smaller the distance, the larger the IoU value.

In this paper, we will traverse different types of labeled boxes in the dataset and cluster different types of boxes. Some specific parameters in equation (6) are as follows: (x_j, y_j, w_j, h_j), $j \in \{1, 2, \ldots, k\}$, is the coordinates of the label boxes. (x_j, y_j) is the center point of the box, (w_j, h_j) is the width and height of the boxes, and N is the number of all label boxes. Given k cluster center points (W_i, H_i), $i \in \{1, 2, \ldots, k\}$, where W_i and H_i are the width and height of the prior bounding box. Calculate the distance between each label box and each cluster center, and the center of each label box coincides with the cluster center during calculation. In this way, the label box is assigned to the nearest cluster center. After all the label boxes are allocated, the cluster centers are recalculated for each cluster. The equation is as follows:

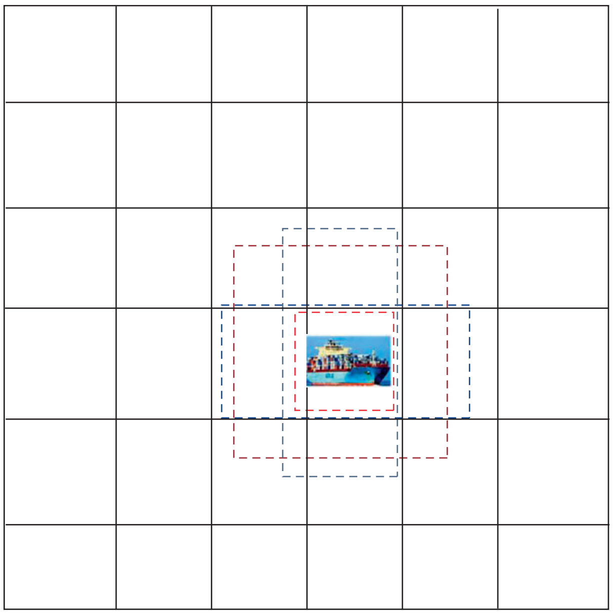


Figure 7: Schematic diagram of the prior bounding box. At this time, the aspect ratio $a_r = \{1, 2\}$, and there are 4 prior bounding boxes.

Table 3: The specific parameters of prior bounding boxes in the SSD algorithm.

Feature map	Size	Numbers	a_r
Conv4_3	38×38	4	1, 2
FC7	19×19	6	1, 2, 3
Conv8_2	10×10	6	1, 2, 3
Conv9_2	5×5	6	1, 2, 3
Conv10_2	3×3	4	1, 2
Conv11_2	1×1	4	1, 2

$$W'_i = \frac{1}{N_i} \sum w_i, \qquad H'_i = \frac{1}{N_i} \sum h_i, \tag{7}$$

where N_i is the number of label boxes in the ith cluster, that is, find the average of all label boxes in the cluster. Repeat the above steps until the cluster center changes very little.

In this paper, we set the number of cluster center $k = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$ to conduct experiments and use the average IoU to measure the results of the experiment, so as to complete the reconstruction of the prior box. It can be seen from Figure 8 that when $k \leq 6$, the average IoU increases greatly, and when $k > 6$, it basically tends to be flat. By combining the calculation amount of the entire algorithm for comprehensive consideration, we choose $k = 6$. At this time, the aspect ratio of the prior bounding box is predicted to be [0.35, 0.89, 1.18, 1.69, 1.89, 2.86]. Table 4 shows the specific parameters of the prior bounding box setting in the NSD-SSD algorithm. Through the method of prior bounding box reconstruction, the error of the algorithm is reduced with improved accuracy and efficiency.

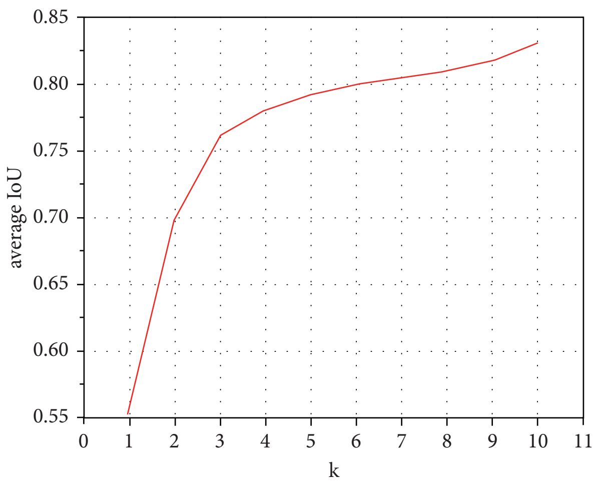


Figure 8: The clustering map of the prior bounding box.

3.3. *Loss Function.*

When training the detection network, we need to measure the error between the candidate boxes and the truth value boxes and minimize this error. At this time, for each candidate box, the offset of the center point of the candidate box relative to the center of the truth box and the confidence of the candidate box needs to be calculated. In the training phase, there are generally two samples, called positive samples and negative samples. Here, we consider the matching value of the candidate box and the truth box to be

Table 4: The specific parameters of prior bounding boxes in the NSD-SSD algorithm.

Feature map	Size	Numbers	a_r
Conv4_3	38×38	6	1, 2, 3
FC7	19×19	6	1, 2, 3
Conv8_2	10×10	6	1, 2, 3
Conv9_2	5×5	6	1, 2, 3
Conv10_2	3×3	6	1, 2, 3
Conv11_2	1×1	6	1, 2, 3

greater than the threshold as positive samples, denoted by d^1, and other candidate boxes that do not satisfy minimum matching value are considered negative samples, denoted by d^2. In order to ensure the balance of the sample, the ratio of positive and negative samples is required to be at most 3: 1.

The loss function of the NSD-SSD algorithm is basically similar to that of the SSD. In this study, the total loss function includes the classification loss and the localization loss:

$$L(x,c,l,g) = \frac{1}{N}\left(L_{\text{cls}}(x,c) + \alpha L_{\text{loc}}(x,l,g)\right), \quad (8)$$

where N is the number of the positive samples. If $N = 0$, we set the loss to 0. c is confidence, l is the predicted box, and g is the ground truth box. α is the balance coefficient between classification loss and localization loss, and its value usually is 1.

The localization loss is smooth $L1$ loss, x_{ij}^p is an indicator, and $x_{ij}^p = \{0,1\}$. When $x_{ij}^p = 1$, it means that the ith candidate box matches jth ground truth box of ship category p:

$$L_{\text{loc}}(x,l,g) = \sum_{i\in d^1}^{N} \sum_{m\in\{cx,xy,w,h\}} x_{ij}^k \text{smooth}L1\left(l_i^m - \widehat{g}_j^m\right), \quad (9)$$

where

$$\text{smooth}L1(x) = \begin{cases} 0.5x^2 & |x| < 1, \\ |x| - 0.5 & \text{otherwise.} \end{cases} \quad (10)$$

The classification loss is the Softmax loss. When classifying, the confidence level belonging to the ship category p is expressed by c^p, and the confidence level belonging to the background is expressed as c^0:

$$L_{\text{cls}}(x,c) = -\sum_{i\in d^1}^{N} x_{ij}^p \log\left(\widehat{c}_i^p\right) - \sum_{i\in d^2} \log\left(\widehat{c}_i^0\right), \quad (11)$$

where $\widehat{c}_i^p = \exp(c_i^p)/\sum_p \exp(c_i^p)$. In the first half of equation (11), the predicted frame i and the real frame j match with respect to the ship category p. The higher the predicted probability of p, the smaller the loss. In the second half of the equation, there is no ship in the predicted box. That is, the higher the predicted probability of the background, the smaller the loss. In this study, we use Stochastic Gradient Descent to optimize the loss function to find the optimal solution. The final loss function curve of NSD-SSD is shown in Figure 9. Note that due to the result of deep learning in this model, the loss function in the early stage will fluctuate, but it will eventually become stable.

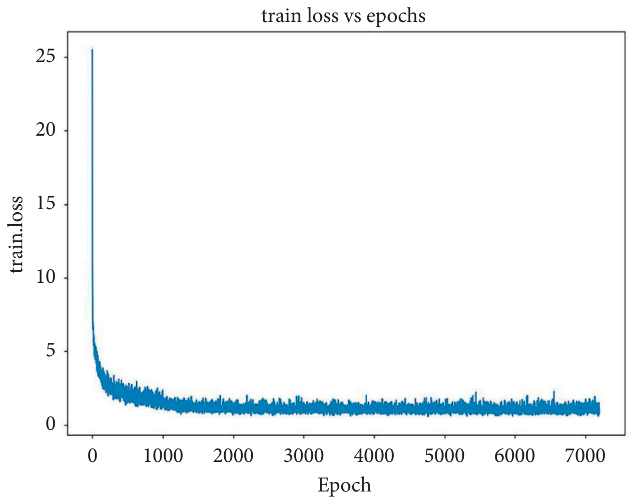


Figure 9: The loss function curve of the NSD-SSD.

4. Experimental Results

To prove the effectiveness of our proposed method, we designed experiments and quantitatively evaluated the proposed method on the public ship dataset. Subjective and objective results will be presented in this section, and the results will also be analyzed.

4.1. Dataset. In this paper, we use a public dataset called SeaShips [44] for ship detection. This dataset consists of 6 common ship categories and 7000 images in total, including ore carrier, bulk cargo carrier, general cargo ship, container ship, fishing boat, and passenger ship. All of the images are video clips taken by surveillance camera, covering all possible imaging changes, with different proportions, hull parts, background, and occlusion. All images are marked with ship category labels and bounding boxes. The example images of each ship category are shown in Figure 10. In order to better train and evaluate the model, we divided the dataset into a training set, a validation set, and a testing set. The 3500 images were randomly selected as the training set, 1750 images as the validation set, and the rest as the testing set. In particular, the validation set was useful to avoid overfitting for better model selection.

4.2. Test Settings. All the models in our experiment are run on a 64-bit Ubuntu operating system using a 2.9 GHz Intel Core-i5 with 15.6 GB of RAM and NVIDIA GTX 1080Ti

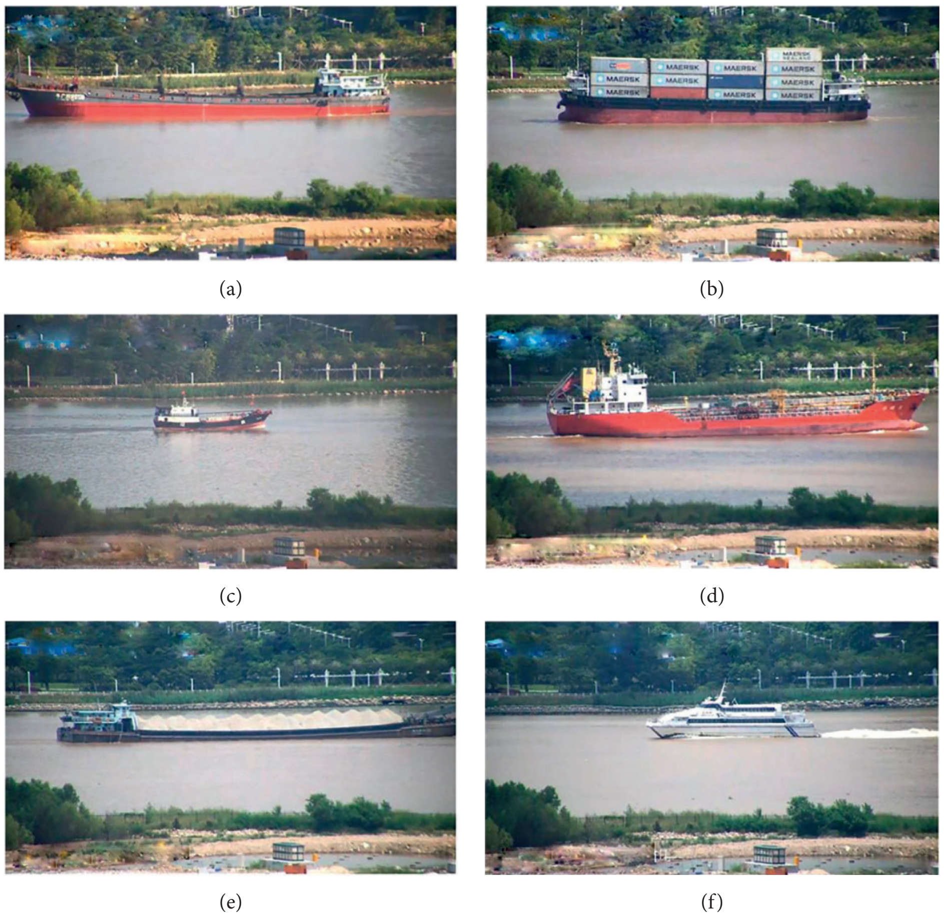


FIGURE 10: The ship category in our used dataset. (a) Bulk cargo carrier. (b) Container ship. (c) Fishing boat. (d) General cargo ship. (e) Ore carrier. (f) Passenger ship.

GPU with 11 GB of video RAM. The deep learning framework that we use is Pytorch which runs on GPU.

Our proposed network structure is modified from SSD, and the NSD-SSD and SSD use the same hyperparameters for training. The batch size we used is 32, and the num_-workers is 4. The initial learning rate is set to 0.001. Momentum is 0.9, and weight decay is 0.0002.

4.3. Evaluation Index. Since this article studies the task of ship object detection, several mature indicators are needed to evaluate the detection model. These indicators will be described in detail below.

(1) Intersection-over-Union (IoU): IoU is a standard for measuring the accuracy of detecting the position of corresponding objects in a specific dataset. In other words, this standard is used to measure the correlation between real and predicted. The higher the correlation, the greater the value. The equation is as follows:

$$\mathrm{IoU} = \frac{G_t \cap D_r}{G_t \cup D_r}. \tag{12}$$

In equation (12), G_t is the ground-truth bounding box, D_r is the predicted bounding box, $G_t \cap D_r$ is the intersection of G_t and D_r, and $G_t \cup D_r$ is the union of G_t and D_r. The range of IoU is 0-1; in this paper, we set the threshold to 0.5. Once the IoU calculation result is greater than 0.5, it is marked as a positive sample; otherwise, it is also a negative sample.

(2) Average precision: After the IoU threshold is given, there will be two indicators called precision and recall. The precision refers to the number of ground truth ships in all predictions. The recall refers to the number of ground truth ships predicted in all ground truth ships. So, precision and recall are as follows:

$$\begin{aligned} \text{precision} &= \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}, \\ \text{recall} &= \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}. \end{aligned} \tag{13}$$

According to precision and recall, a precision-recall curve can be drawn, referred to as the PR curve. AP is the area enclosed by this curve, and the specific equation is as follows:

$$\mathrm{AP} = \int_0^1 P(R)\mathrm{d}R, \tag{14}$$

(3) mean Average Precision (mAP): mAP shows the average values of AP_i of each class i:

$$\mathrm{mAP} = \frac{\sum_{i=1}^{n} \mathrm{AP}_i}{n}. \tag{15}$$

Here, n represents the number of classes of ships that need to be detected.

(4) *Frames Per Second (FPS)*: the FPS is used to judge the detection speed of different models, and the larger the FPS value, the faster the speed.

4.4. Results and Analysis. Our model is based on the SSD network with the backbone of VGG16. To test the detection performance of NSD-SSD, the comparative experiments are implemented using several popular baseline methods: Faster R-CNN, SSD series, and YOLO series. The backbone networks of these models are pretrained on ImageNet. To achieve a fair comparison, we train and test the four models using the same dataset. At the same time, to ensure the consistency of training, we set the hyperparameters and the number of training echoes of these three baseline models to be the same as the NSD-SSD.

According to our detection method (NSD-SSD), the AP performance of the six categories of ship is shown in Figure 11. The IoU threshold is set to 0.5 in the experiment.

We record the accuracy of the four models based on the evaluation indicators, as shown in Table 5. The detection performance of Faster R-CNN is significantly better than YOLO series and SSD series. On average, Faster R-CNN's mAP is 22.5% and 17.8% higher than SSD series and 12.5% and 8.2% higher than YOLO series, respectively. Although our proposed model (NSD-SSD) has a little gap with Faster R-CNN in mAP, our approach significantly improves the performance of SSD. Moreover, it performs better than Faster R-CNN on general cargo ship.

Our proposed method is based on SSD (VGG16). The detection effect of the original SSD network is indeed not good, and the accuracy is extremely average. But compared with SSD, the mAP of each category of ship in our model has a good improvement and the NSD-SSD's mAP is 20.2% higher than original SSD. Among six categories of ships, the container ships have the best detection results. Because they mainly transport containers, and these cargoes have very distinct shape characteristics that are different from other ships. The ore carriers also achieve excellent detection results. Because they usually transport ore, they have the special features like container ships. In addition, since the general cargo ships are very large in the images, their results are also extremely good. On the contrary, the performance of fishing boats is the worst among these six categories of ships. The main season is that fishing boats are too small, occupying a few pixels in the 1920 × 1080 image. Detectors are generally not good at detecting small objects. After layers of convolution, the feature information of small objects will become blurred, and even the SSD model is worse.

We perform structural improvements on the basis of SSD and add detailed detection techniques, which makes it possible for us to better detect small targets and improve the overall accuracy. For fishing boats, we have increased from 60.4% to 82.4%, which already exceeds the mAP of the YOLOv3 model. As shown in examples in Figure 12, our proposed method greatly improves the detection effect of fishing boats against SSD. For the passenger ships, our method has increased by nearly 10%. For the general cargo ships, our method makes their performance become better and has a significant improvement over Faster R-CNN.

In terms of detection speed, FPS of 24 is called the standard for real-time detection in object detection. As can be seen from Table 6, the detection speed of YOLOv3 is much better than other detection model, and the FPS reaches 79. Unfortunately, its detection effect is not good. The detection speed of SSD series can be ranked second, and the FPS can reach 75 and 68.0, respectively, but detection performance is worse. Since the Faster R-CNN is a two-stage detector, the detection process is more complicated, which results in its FPS of only 7 and cannot meet real-time detection. Our proposed model adds many parameters and calculations on the basis of SSD, thereby reducing the speed. The FPS given by our method is 45, which not only guarantees the real-time detection requirements but also improves the detection accuracy. In addition, we also give the parameters of IoU and recall for different models, and our method is better than other methods.

In Figure 13 we show the detection examples of our model against Faster R-CNN and YOLOv3, and our proposed method has a better visual effect. Specifically, when the two ships are very close together, the bounding box of YOLOv3 is much larger or smaller than ship, but our method can mark a more accurate box. Furthermore, the Faster R-CNN sometimes detects the background as a ship, but our proposed method can avoid the false detection.

We compare the proposed method with [35], and they propose a detection method that combines YOLOv2 and saliency detection and achieve good results. The comparison results are shown in Table 7. From the table, our method is slightly better than the comparison method on mAP. Among the six categories of ships, container ships and fishing boats can achieve better results. Specifically, these two categories of ships' AP is 7.7% and 4.1% higher than the comparison method, respectively. For passenger ships, our method is 3.8% lower than Shao's method because the color characteristic of passenger ships is very salient, the performance of their proposed saliency detection is particularly good, and the accuracy is higher. In addition, the IoU of our method is higher and the detection visual effect is better, but the FPS of Shao's model is 4 higher than the FPS of our model.

To verify the effectiveness of our proposed various modules, we conduct the ablation experiment for comparison, and the original SSD is the baseline network. Moreover, our proposed three modules are considered as setting items, and the experimental results are shown in Table 8.

As can be seen from the table that the detection accuracy of SSD is 10.7% higher than that of the backbone network VGG16, indicating that SSD is a better detection network.

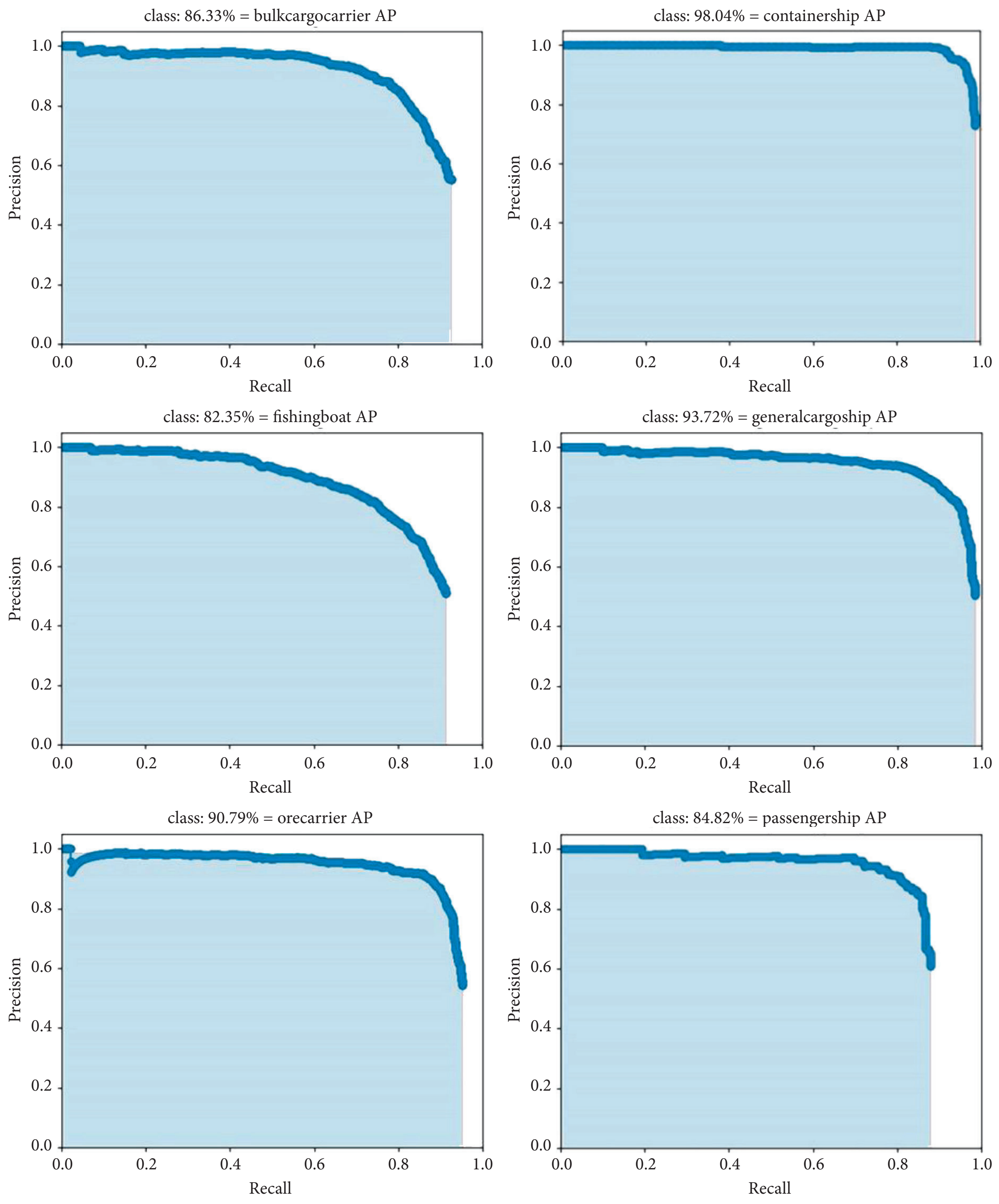


Figure 11: Precision-recall curves of our proposed method (NSD-SSD) on six categories of ships.

When introducing the feature fusion in SSD, the mAP has increased from 69.1% to 83.2%. Because our algorithm considers the combination of shallow features and deep features and makes full use of contextual information. When adding the remaining two parts of modules, the mAP has increased by 6.1%. The above results prove that our proposed method can effectively improve the accuracy of ship detection.

Furthermore, we validate our proposed method under practical extreme conditions, as shown in Figure 14, and under different weather conditions, such as sunny, rainy, and night. On the contrary, the ships in the images are incomplete. However, our method still achieves excellent detection performance, and the marked bounding boxes and the classifications are reasonable and accurate, respectively.

Table 5: Detection accuracy of different detection models.

Model	mAP	Bulk cargo ship	Container ship	Fishing boat	General cargo ship	Ore carrier	Passenger ship
Faster R-CNN	**0.916**	**0.893**	**0.986**	**0.908**	0.927	**0.914**	**0.868**
SSD (VGG16)	0.691	0.661	0.801	0.604	0.703	0.620	0.755
SSD (Mobilev2)	0.738	0.703	0.876	0.635	0.742	0.686	0.783
YOLOv3	0.791	0.681	0.959	0.690	0.893	0.734	0.786
YOLOv4	0.834	0.849	0.929	0.732	0.851	0.778	0.862
NSD-SSD	0.893	0.863	0.980	0.824	**0.937**	0.908	0.848

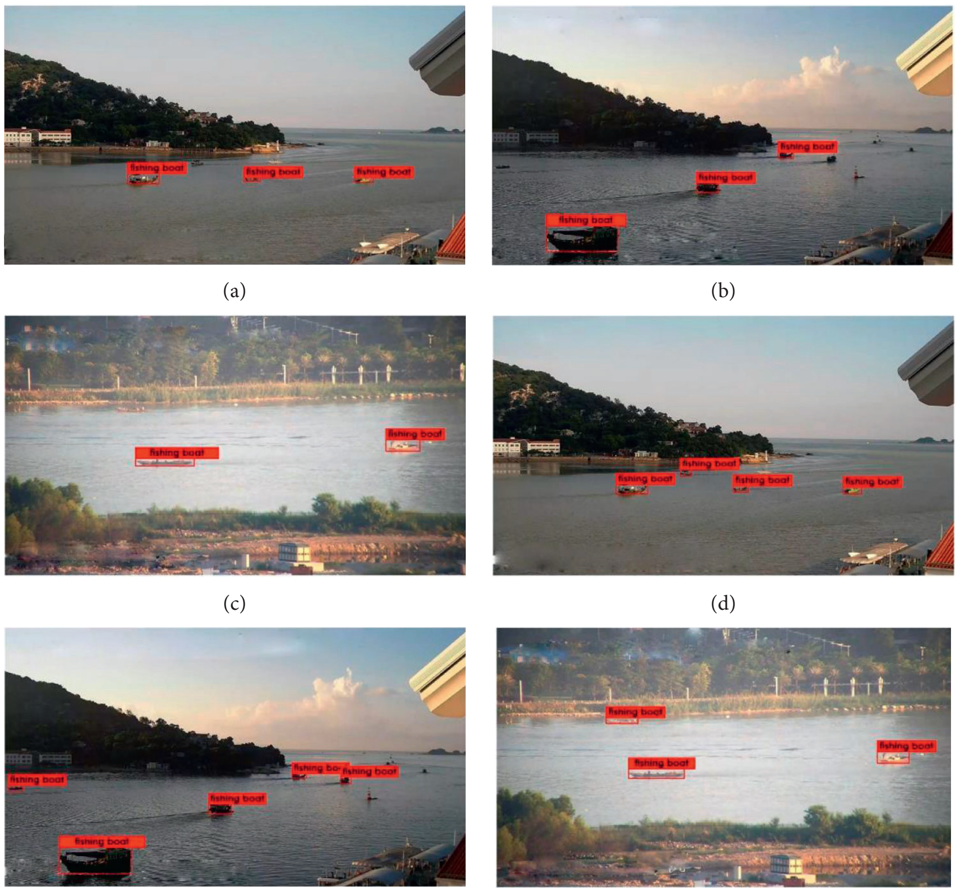


Figure 12: Some fishing boats' detection results. (a–c) The original SSD. (d–f) Our proposed method.

Table 6: The detection results of other indicators for different detectors.

Model	IoU	Recall	FPS
Faster R-CNN	0.603	0.865	7
YOLOv3	0.616	0.834	**79**
SSD (VGG16)	0.781	0.700	75
SSD (Mobilev2)	0.745	0.787	68
YOLOv4	0.716	0.854	56
Ours	**0.808**	**0.936**	45

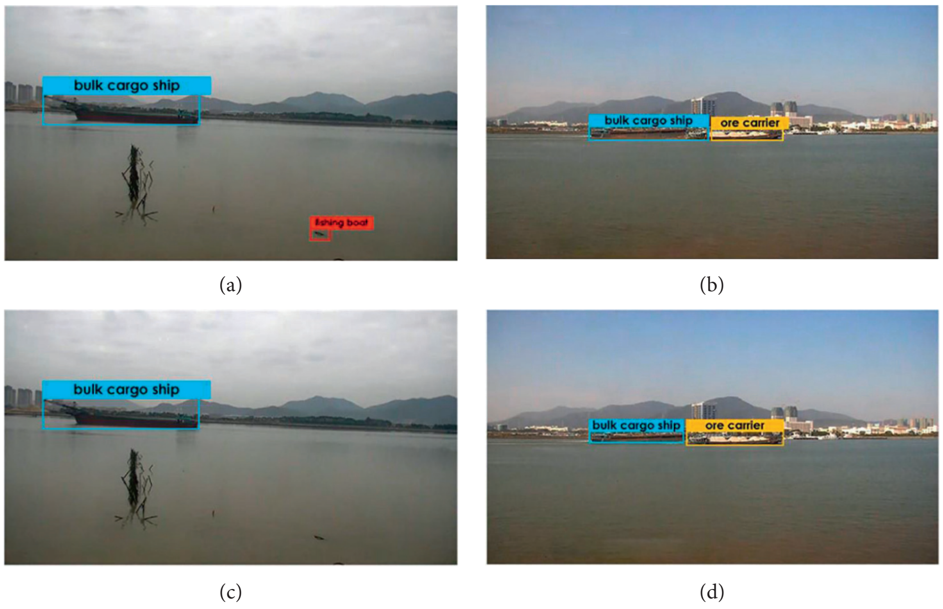


Figure 13: Ship detection results. (a) The Faster R-CNN. (b) YOLOv3. (c, d) Our proposed model.

Table 7: Detection results of different detection models.

Model	IoU	mAP	Bulk cargo ship	Container ship	Fishing boat	General cargo ship	Ore carrier	Passenger ship	FPS
Ours	**0.8082**	**0.893**	0.863	**0.980**	**0.824**	**0.937**	**0.908**	0.848	45
Shao's	0.7453	0.874	**0.876**	0.903	0.783	0.917	0.881	**0.886**	**49**

Table 8: The results of the ablation experiment.

VGG16	SSD	Feature fusion	Improved predicted module	Prior boxes reconstruction	mAP
✓					0.584
	✓				0.691
	✓	✓			0.832
	✓	✓	✓	✓	**0.893**

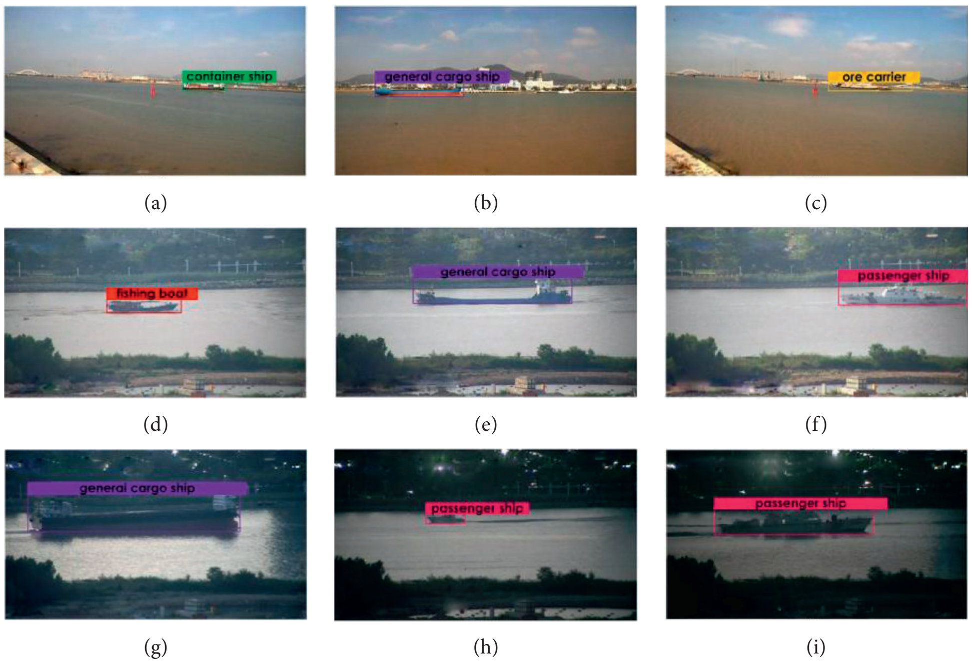


Figure 14: Continued.

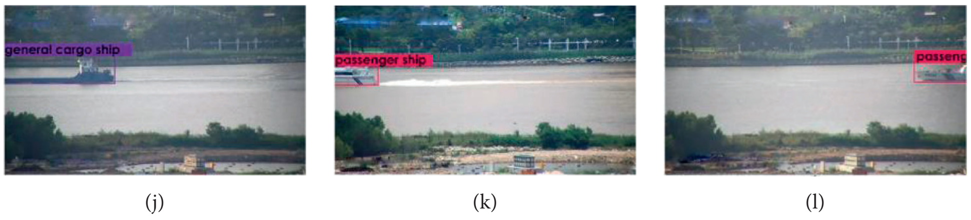


Figure 14: Visualization of ship detection with the proposed method under different conditions. (a–c) Sunny. (d–f) Rainy. (g–i) Night. (j–l) Incomplete ship.

5. Conclusion

In this paper, based on real-time ship detection task as our basic goal as well as the characterization of the ship dataset, a novel ships' detector in visual images captured by the monitoring sensor, named NSD-SSD, is proposed. The NSD-SSD is mainly based on multiscale feature fusion (MFF), predicted module (PM), and reconstruction of prior boxes (RPB). Regarding the problem of small objects detection, the dilated convolution is used to expand the receptive field of low-level feature layers, and the network can fully use the contextual information by the MFF. For the problem of setting prior boxes manually, we propose RPB by using the *K*-means clustering algorithm to improve the detection efficiency. In addition, the PM is introduced to extract deeper features. We train our model on the ship dataset and compare it with other conventional methods. The experimental results prove that our proposed method is able to acquire higher accuracy and recall, and it can meet the requirement of real-time detection. Moreover, the NSD-SSD can also guarantee high-quality detection performance in the relatively extreme environment. We also noticed that the method could be improved for ship detection in complex backgrounds. We will address this issue in our future work.

Data Availability

SeaShip: http://www.lmars.whu.edu.cn/prof_web/shaozhenfeng/datasets/SeaShips(7000).zip (accessed on 2 November 2020).

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was funded by the National Key Research and Development Project of China (Grant no. 2019YFB1600605).

References

[1] S. Dugad, V. Puliyadi, H. Palod, N. Johnson, S. Rajput, and S. Johnny, "Ship intrusion detection security system using image processing & SVM," in *Proceedings of the 2017 International Conference on Nascent Technologies in Engineering (ICNTE)*, pp. 1–7, Navi Mumbai, India, 2017.

[2] X. Cao, S. Gao, L. Chen, and Y. Wang, "Ship recognition method combined with image segmentation and deep learning feature extraction in video surveillance," *Multimedia Tools and Applications*, vol. 79, no. 13-14, pp. 9177–9192.

[3] L. Jiao, F. Zhang, F. Liu et al., "A survey of deep learning-based object detection," *IEEE Access*, vol. 7, pp. 128837–128868, 2019.

[4] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proceedings of the Advances in Neural Information Processing Systems 30 (NIPS 2017)*, pp. 1097–1105, Lake Tahoe, NV, USA, December 2012.

[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015, https://arxiv.org/abs/1409.1556.

[7] C. Szegedy, W. Wei Liu, Y. Sermanet 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, USA, 2015.

[8] 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, 2016.

[9] G. Huang, Z. Liu, and L. Weinberger, "Densely connected convolutional networks," in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2261–2269, Honolulu, HI, USA, 2017.

[10] A. Abdollahi, B. Pradhan, S. Gite, and A. Alamri, "Building footprint extraction from high resolution aerial images using generative adversarial network (GAN) architecture," *IEEE Access*, vol. 8, pp. 209517–209527, 2020.

[11] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 580–587, Columbus, OH, USA, 2014.

[12] 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.

[13] R. Girshick, "Fast R-CNN," in *Proceedings of the International Conference on Computer Vision (ICCV)*, Santiago, Chile, 2015.

[14] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017.

[15] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 2, pp. 386–397, 2020.

[16] P. Sermanet, D. Eigen, and X. Zhang, "OverFeat: integrated recognition, localization and detection using convolutional networks," 2013, https://arxiv.org/abs/1312.6229.

[17] W. Liu, D. Anguelov, D. Erhan et al., "SSD: single shot multibox detector," in *Proceedings of the European Conference on Computer Vision*, pp. 21–37, Springer, Amsterdam, The Netherlands, October 2016.

[18] J. Redmon, S. Divvala, and R. Girshick, "You only look once: unified, real-time object detection," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 779–788, Las Vegas, NV, USA, June 2016.

[19] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6517–6525, Honolulu, HI, USA, 2017.

[20] J. Redmon and A. Farhadi, "YOLOv3: an incremental improvement," 2018, https://arxiv.org/abs/1804.02767.

[21] A. Bochkovskiy, C. Wang, and H. Liao, "YOLOv4: optimal speed and accuracy of object detection," 2020, https://arxiv.org/abs/2004.10934.

[22] T. Yulin, S. Jin, G. Bian, and Y. Zhang, "Shipwreck target recognition in side-scan sonar images by improved YOLOv3 model based on transfer learning," *IEEE Access*, vol. 8, pp. 173450–173460, 2020.

[23] Z. Chen, K. Wu, Y. Li, M. Wang, and W. Li, "SSD-MSN: an improved multi-scale object detection network based on SSD," *IEEE Access*, vol. 7, pp. 80622–80632, 2019.

[24] C. Kang and X. Zhang, "A fusion algorithm of multiple classifiers for recognition of ships radiated_noises based on many-person decision-makings theory," in *Proceedings of the Fourth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2007)*, pp. 71–74, Haikou, China, 2007.

[25] C. Zhao, L. Zhengguo, S. Xiaohong, B. Jun, and L. Zhengguo, "A decision tree SVM classification method based on the construction of ship-radiated noise multidimension feature vector," in *Proceedings of the 2011 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*, pp. 1–6, Xi'an, China, 2011.

[26] J. Luo and Y. Wang, "Recognition and location technique of volume target based on frequency domain characteristics of ship radiated noise," in *Proceedings of the 2011 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*, pp. 1–5, Xi'an, China, 2011.

[27] C. Peng, L. Yang, X. Jiang, and Y. Song, "Design of a ship radiated noise model and its application to feature extraction based on winger's higher-order spectrum," in *Proceedings of the 2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, pp. 582–587, Chengdu, China, 2019.

[28] C. Zhu, H. Zhou, R. Wang, and J. Guo, "A novel hierarchical method of ship detection from spaceborne optical image based on shape and texture features," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 48, no. 9, pp. 3446–3456, 2010.

[29] G. Ge Liu, Y. Yasen Zhang, and X. Xian Sun, "A new method on inshore ship detection in high-resolution satellite images using shape and context information," *IEEE Geoscience and Remote Sensing Letters*, vol. 11, no. 3, pp. 617–621, 2014.

[30] Z. Zhenwei Shi, X. Xinran Yu, and Z. Bo Li, "Ship detection in high-resolution optical imagery based on anomaly detector and local shape feature," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 8, pp. 4511–4523, 2014.

[31] W. Wenxiu, F. Yutian, and D. Feng, "Remote sensing ship detection technology based on DoG preprocessing and shape features," in *Proceedings of the 2017 3rd IEEE International Conference on Computer and Communications (ICCC)*, pp. 1702–1706, Chengdu, China, 2017.

[32] Y. Zou, L. Zhao, S. Qin, M. Pan, and Z. Li, "Ship target detection and identification based on SSD_MobilenetV2," in *Proceedings of the 2020 IEEE 5th Information Technology and Mechatronics Engineering Conference (ITOEC)*, pp. 1676–1680, Chongqing, China, 2020.

[33] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: inverted residuals and linear bottlenecks," in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, Salt Lake City, UT, USA, 2018.

[34] Z. Lin, K. Ji, X. Leng, and G. Kuang, "Squeeze and excitation rank faster R-CNN for ship detection in SAR images," *IEEE Geoscience and Remote Sensing Letters*, vol. 16, no. 5, pp. 751–755, 2019.

[35] Z. Shao, L. Wang, Z. Wang, W. Du, and W. Wu, "Saliency-aware convolution neural network for ship detection in surveillance video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 3, pp. 781–794, 2020.

[36] X. Nie, M. Duan, H. Ding, B. Hu, and E. K. Wong, "Attention Mask R-CNN for ship detection and segmentation from remote sensing images," *IEEE Access*, vol. 8, pp. 9325–9334, 2020.

[37] B. Guo, J. Shi, L. Zhu, and Z. Yu, "High-speed railway clearance intrusion detection with improved SSD network," *Applied Sciences*, vol. 9, no. 15, p. 2981, 2019.

[38] Z. Huang, B. Sui, J. Wen, and G. Jiang, "An intelligent ship image/video detection and classification method with improved regressive deep convolutional neural network," *Complexity*, vol. 2020, Article ID 1520872, 11 pages, 2020.

[39] Y. Zhao, L. Zhao, B. Xiong, and G. Kuang, "Attention receptive pyramid network for ship detection in SAR images," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 13, pp. 2738–2756, 2020.

[40] Z. Li, Y. You, and F. Liu, "Analysis on saliency estimation methods in high-resolution optical remote sensing imagery for multi-scale ship detection," *IEEE Access*, vol. 8, pp. 194485–194496, 2020.

[41] M. Zhao, Y. Zhong, and D. Sun, "Accurate and efficient vehicle detection framework based on SSD algorithm," *IET Image Process*, pp. 1–11, 2021.

[42] Z. Cai, Q. Fan, R. S. Feris, and N. Vasconcelos, "A unified multi-scale deep convolutional neural network for fast object detection," in *Proceedings of the 14th European Conference on Computer Vision, ECCV 2016*, pp. 354–370, Amsterdam, The Netherlands, October 2016.

[43] C. Fu, W. Liu, and A. Ranga, "DSSD: deconvolutional single shot detector," 2017, https://arxiv.org/abs/1701.06659.

[44] Z. Shao, W. Wu, Z. Wang, W. Du, and C. Li, "SeaShips: a large-scale precisely annotated dataset for ship detection," *IEEE Transactions on Multimedia*, vol. 20, no. 10, pp. 2593–2604, 2018.

Hindawi
Computational Intelligence and Neuroscience
Volume 2021, Article ID 7367870, 15 pages
https://doi.org/10.1155/2021/7367870

Research Article

Anomaly Detection in Videos Using Two-Stream Autoencoder with Post Hoc Interpretability

Jiangfan Feng,[1] Yukun Liang,[1] and Lin Li[2]

[1]*Chongqing University of Posts and Telecommunications, School of Computer Science and Technology, Chongqing, China*
[2]*Chongqing Geomatics and Remote Sensing Center, Chongqing, China*

Correspondence should be addressed to Jiangfan Feng; fengjf@cqupt.edu.cn

Received 29 April 2021; Revised 16 June 2021; Accepted 14 July 2021; Published 27 July 2021

Academic Editor: Francisco Gomez-Donoso

The growing interest in deep learning approaches to video surveillance raises concerns about the accuracy and efficiency of neural networks. However, fast and reliable detection of abnormal events is still a challenging work. Here, we introduce a two-stream approach that offers an autoencoder-based structure for fast and efficient detection to facilitate anomaly detection from surveillance video without labeled abnormal events. Furthermore, we present post hoc interpretability of feature map visualization to show the process of feature learning, revealing uncertain and ambiguous decision boundaries in the video sequence. Experimental results on Avenue, UCSD Ped2, and Subway datasets show that our method can detect abnormal events well and explain the internal logic of the model at the object level.

1. Introduction

The video's abnormal event detection is to find events different from usual, such as people fighting or urgent events like fire. It is an essential task in the computer vision field, from both academia and industry. As video cameras continue to expand, exploiting video data is currently severely limited by the amount of human effort, so that the automatic detection of rare or unusual incidents and activities in a surveillance video is urgently needed [1]. Although abnormal event detection has inspired plenty of works based on computer vision techniques [2–4], it is still quite challenging to design a general detection framework because of the definition uncertainty and limitations of the data-generating mechanism.

Deep learning technologies have been widely used to detect abnormal events, including unsupervised methods [5, 6] and weakly supervised methods [7]. Recently, another developing approach for video processing in the deep learning framework is two-stream networks, which have been successfully applied to video-based action recognition [1, 4], often with state-of-the-art results. Despite the excellent performance, none of these methods considers the black-box problem brought by deep learning models. Regarding practical use, to ensure the proposed method can produce reliable results, a model's interpretability is required.

Advances in computer vision have led to an interest in automated computational methods for video surveillance. The anomaly detection task [5–12] mainly trains a regular model with only normal samples and then marks the samples in the test dataset different from the normal samples. Recently, more and more approaches have employed deep learning [5–7, 12–20] to learn the features of the video frame. Those methods train the model to get better detection results. For instance, Ravanbakhsh et al. [17] proposed combining CNN high-level semantic information and low-level Optical-Flow as a new method of measuring anomalies.

Besides, Feichtenhofer et al. [3] proposed to use SlowFast Networks for video recognition. SlowFast networks can be described as a single stream architecture that operates at two different frame rates. In contrast, we use the two-stream autoencoder network to learn features and generate the reconstructed video sequence to detect anomalies. Tudor Ionescu et al. [10] first proposed to use unmasking to deal with learned characteristics. Different from our method, it is to use a binary classifier to determine anomalies. Fan et al. [11] proposed the Gaussian Mixture Variational Autoencoder, which used the Gaussian Mixture Model (GMM) to fit the distribution of the feature space through the variational method. Sabokrou et al. [13] proposed a cubic-patch-based way containing 3D deep autoencoders and 3D convolutional neural networks for an advanced feature-learning approach. Luo et al. [18] proposed to combine Convolutional Long Short-Term Memory (Conv-LSTM) with Autoencoder to learn the appearance and action information of the video. The model will output the reconstructed sequence input at the current time and last time. Chong and Tay [20] proposed an effective method for video anomaly detection, which is suitable for the spatiotemporal structure of video anomaly detection, including crowded scenes.

Herein, we propose an unsupervised learning scheme to detect abnormal events using a novel two-stream network by utilizing late fusion, with its inherent logic through post hoc interpretability: (1) We propose an abnormal event detection algorithm in surveillance video that offers a potential improvement on two key elements, that is, the interpretability and the performance of detection, which is of great significance in video surveillance. (2) The proposed two-stream architecture learns the appearance characteristics of the video through a spatial model, and the temporal model is a temporal autoencoder to learn the regular pattern in the video. The advantage is the suitability of modeling the spatial characteristics using relatively few training samples. (3) We visualize the feature map of the convolution layers and outputs the features learned by the convolution layers at the object level through a heatmap, enabling abnormal object detection. Furthermore, it helps users identify essential features of surveillance tasks, demonstrate the importance of features, and reproduce the decisions made by the black-box model.

2. Related Work

Applying the method of interpretable deep learning to anomaly detection is an emerging research direction, and it is still in the development stage. The extension includes two significant aspects.

2.1. Semantics of the CNN. Although CNNs have achieved significant momentum in computer vision tasks, the end-to-end learning strategy brings about infrequent interpretability [21]. On the other hand, it can help ensure impartiality in decision-making and provides a truthful causality in model inference [22]. The visualization of filters in CNNs is the most direct way to explore the visual patterns hidden in neural units. Firstly, most visualizations are gradient-based methods [23–27]. These methods mainly calculated the gradient scores of convolutional neural network units and used them to evaluate the image's appearance to maximize its unit fraction. Similar approaches, up-convolutional networks [28], were a typical way of visualizing the representations of CNNs. Besides, Zhou et al. [29] provided a method to precisely calculate the neural activation image receptive field. However, these methods are postinterpretation of online learning, which did not adjust the model or affect the final decision.

Apart from neural network visualization methods, machine learning models can also explain neural networks. Some approaches focused on learning networks with disentangled representations to represent the semantic hierarchy hidden inside CNNs [30, 31]. Zhang et al. [32] proposed a quantitative interpretation of convolutional networks' prediction logic through decision trees. This method can learn explicit representations of object parts in the high convolutional layers of CNNs while mining potential decision modes in fully connected layers. Besides, Zhang et al. [33] proposed modifying CNNs by adding a loss to each filter of a high convolutional layer to receive the deentanglement representation. Wu et al. [34] proposed an interpretable localized convolutional neural network for object detection. These interpretation methods are different from network visualization. In particular, a previous study [35, 36] showed the potential of interpretable deep learning techniques for predicting properties of simulated low-dimensional magnetic systems.

2.2. Abnormal Event Detection. Previous studies in abnormal event detection have suggested that the detection model can be trained from the reconstruction task. Hasan et al. [7] introduced a full autoencoder with manually annotated data, and anomaly detection was based on reconstruction loss. Luo et al. [16] used time-coherent sparse coding to encode two adjacent frames with similar reconstruction coefficients. However, the abnormal events observed in these models were primarily dependent on reconstruction error. As a result, it might fit abnormal events unexceptionally. Thus, the prediction model compared the predicted frame with the actual video frames for anomaly detection. GANs are usually used to enhance the predictive ability [37–40]. Moreover, constraints in motion and gradient are also proven effective. Liu et al. [41] proposed a framework based on future frame prediction to detect anomalies. However, the prediction method can be sensitive to noise and perturbation, especially in scenes with illumination changes, leading to inferior robustness in anomaly detection. Thus, Qiang et al. [42] proposed an anomaly detection model based on the latent feature space, combining the above two methods. In addition to detecting abnormal events from learning-based techniques, Yu et al. [43] proposed a neuromorphic vision sensor, a natural motion detector for abnormal objects. Recently, several authors have presented abnormal video detection by the two-stream convolutional network. Simonyan and Zisserman [44] proposed a two-stream network to recognize the actions of video objects. Kingma and Welling [22] proposed a new fusion method based on the two-stream structure to identify the action information in the video. They found that spatial and temporal networks can be fused in the convolutional layer but not in the softmax layer. Subsequently, Yan et al. [1] proposed a two-stream abnormal detection method, and the model is composed of an appearance stream and action stream.

However, most of the video abnormal event detection algorithms cannot achieve online monitoring. The first difficulty is that the model has many layers, the structure is more complex, and detecting anomalies is too time-consuming. Therefore, we want to learn spatial-temporal features through the two-stream network and learn features through some relatively lightweight architectures, but the detection performance can also be excellent. Besides, none of these deep methods considers the "black-box" characteristics, and it demonstrates the urgent need to apply the internal logic of anomaly detection at the semantic level. Therefore, we want to show the most critical features of surveillance tasks and reproduce the decisions made by the black-box model.

3. Proposed Method

The general workflow for our method (Figure 1) includes two streams (spatial stream and temporal stream), which learn features during the encoding stage, and then generate reconstructed sequences of the raw video sequence through decoding. The method can be considered as unsupervised learning scheme in which an autoencoder is trained on the normal data through reconstruction. If an abnormal event occurs, the corresponding reconstruction error score is higher than the normal data since the model has not met the irregular pattern during training. Besides, we visualized the spatial model's convolutional layer features to identify ways that could help further understand and display the process of model learning at the object level to help people comprehend and trust the detection results of our model.

3.1. Autoencoder-Based Reconstruction. The input to the two-stream network is regular video frames. We trained the model, and the reconstruction error was calculated between the initial and reconstructed frames. Reconstruction error is used to calculate the regularity score that can be further evaluated for the detection performance of the system. Our approach generates reconstruction errors from both the spatial and temporal streams in the testing stage and then fuses them appropriately.

Our approach contains three main steps.

3.1.1. Preprocessing. The various video clips were used to build and test our model, which differed in size, shooting time, and definition. We decomposed the anomaly detection datasets into a sequence of video frames and unified the video frame size to 224×224 pixels. To ensure that the input video frames are all on the same scale, we computed the training image's pixel average. Then, we subtracted each frame from the average global image for normalization. We also converted the image to grayscale to reduce the dimensionality. Because of the large number of learnable parameters and limited given training datasets, we used data augmentation [7] to enlarge the training data set in the temporal dimension. The enlargement is done by generating the new cuboids with various skipping strides to construct T-sized original video frames (for example, In stride-1 cuboids, all T frames are consecutive, whereas, in stride-2 and stride-3, cuboids skip one and two video frames, respectively).

3.1.2. Feature Learning. We used a spatial stream to learn the appearance and used a temporal stream to learn the temporal coherency on adjacent video frames. The temporal model consists of three parts, the convolution layers, the deconvolution layers, and the convolution long short-term memory (Conv-LSTM) layers. The convolutional layer is used to learn each frame's spatial or behavioural characteristics. The deconvolutional layer is used to restore the original input size, and the Conv-LSTM layer outperforms the temporal rules of the video. Our spatial model is similar to the temporal model, but the spatial model lacks a Conv-LSTM layer, and its input is in the form of a single frame instead of consecutive frames.

(1) Spatial Model. Figure 2 shows the detailed configuration of the proposed spatial model. It only consists of three convolutional layers, followed by two deconvolutional layers to improve efficiency. Since anomaly detection focuses more on low-level contours, edge features, the spatial model only uses three convolutional layers for feature extraction. On the other hand, the role of the deconvolutional layer is to generate reconstructed video frames and densify the sparse inputs by operations with multiple filters. Hence, the spatial size of the output feature maps of a deconvolutional layer is larger than the spatial size of its corresponding inputs. Therefore, we extract the person's appearance feature in the video through the three-layer convolution layer and restore the initial input dimensions through the connected two deconvolution layers. The parameters are designed to balance the strength of the convolutional and deconvolutional layers. Therefore, we optimize them alternatively with the layer-parameter set through the training process. During the training stage, the learnable parameters were updated toward the direction minimizing the loss function. We used MSE loss based on the Relu function. By calculating the partial derivatives of the loss function, we could update the parameters in an SGD scheme.

The feature-learning process is the essential stage of model training. In the encoding stage, the model learns the spatial features of the monitored object in the video frame and the critical background information in the monitored scene. Also, the spatial model architecture and input are relatively simple. The feature map visualization algorithm is to transparentize the "black box" of the spatial model, understand the model's learning process, and trust the final detection results.

(2) Temporal Model. The temporal model may have similarly formulated but different layers based on LSTM requirements. To better learn the temporal coherency on adjacent frames, we added three layers of Conv-LSTM between the convolution layers and deconvolution layers (Figure 3). The dimensions of the three layers are the same, and the main difference is the number of convolution kernels.

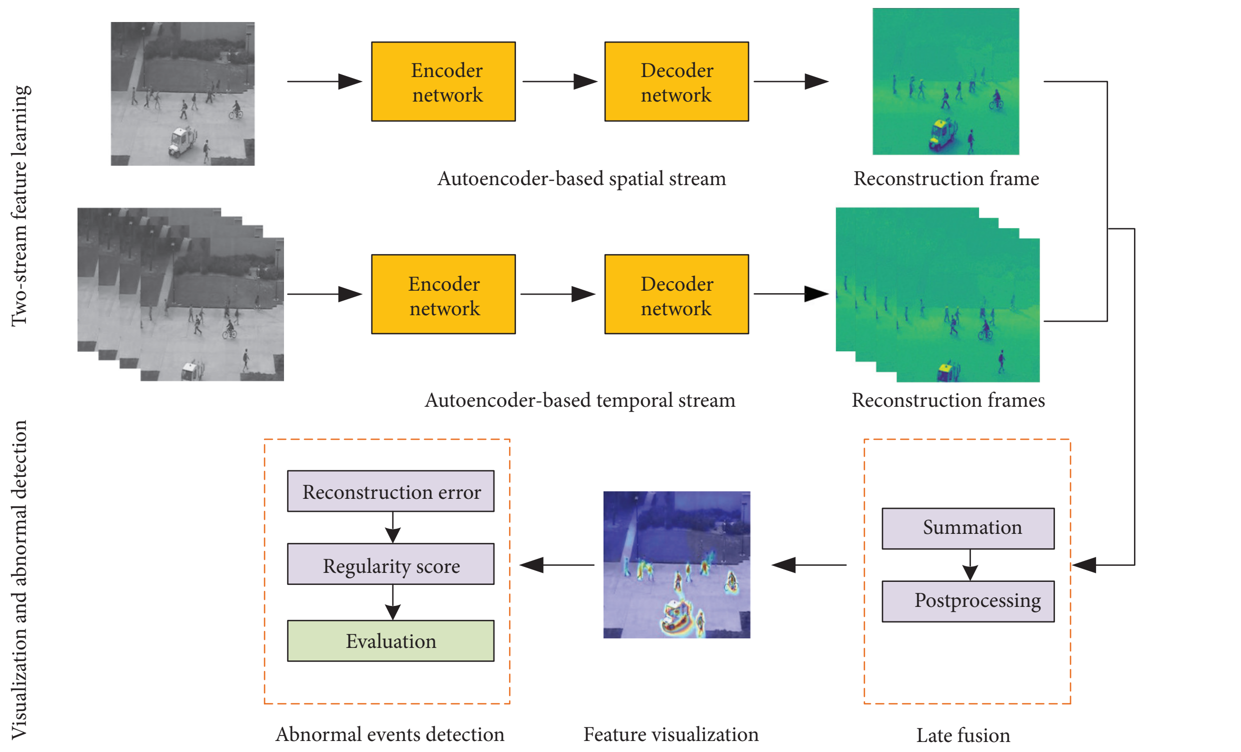


FIGURE 1: The pipeline of our method. First, we need to decompose the video of datasets into continuous frames. Then we use the convolutional two-stream network to learn the feature of image frames. Finally, the video frame's regularity score is calculated to decide this frame is normal or abnormal. From the initial image to the heatmap, we can see that the model is more interested in areas where abnormal events occur, such as the patrol car that appears here.

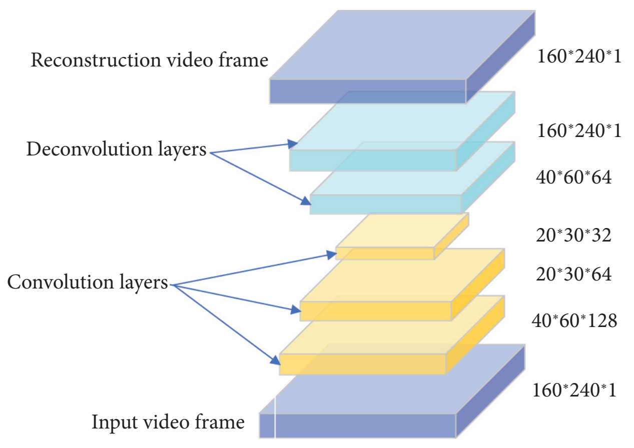


FIGURE 2: Spatial model architecture. The rightmost number indicates the output size of each layer.

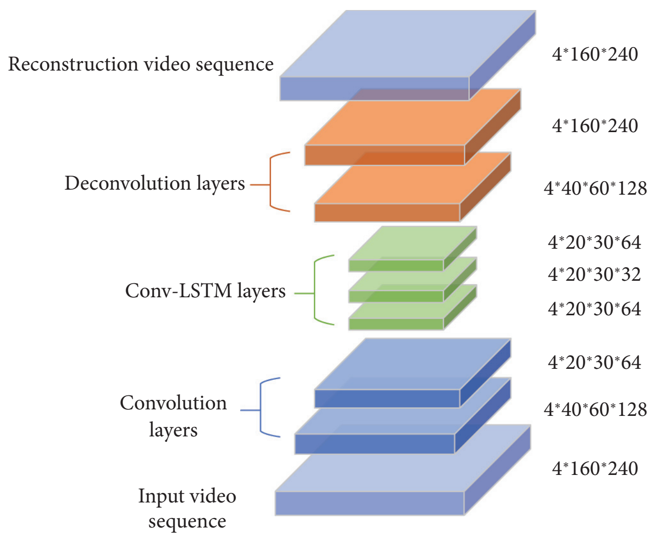


FIGURE 3: Temporal model architecture. The final output is the reconstructed frame sequence.

The input to the temporal model is the video volume. Considering the effect of frame length on model training and memory consumption speed, we chose four consecutive frames with various skipping strides in this paper. The frame number is a trade-off parameter in learning. This length of training on the subway dataset is just what our machine can meet, and the speed of training and testing was relatively good. Shi et al. [45] proposed the Conv-LSTM model first, and Patraucean et al. [46] utilized the model to predict the next video frame. To extract both temporal and spatial features of the Conv-LSTM model, we inputted the image as X, and a convolution filter replaces the set of weights for each connection, which can get the timing relationship and extract the spatial features like the convolutional layer.

3.1.3. Reconstruction Error. After we got the reconstructed sequence of the video frame, we calculated its reconstruction error between the initial video frame and the reconstructed frame to model standard data's probability distribution. In our proposal, reconstruction is a stochastic process that considers the distance between the reconstruction and the initial video frame and the variability of the distribution itself. To qualitatively analyze whether our model can detect anomalies well, we used the regularity score graph to indicate the ability of our model to detect anomalies. The regularity score corresponds to the level of normality of each frame in the video.

In practice, we first counted the reconstruction error of the video frame before getting the regularity score $\mathrm{sr}(t)$. Then, we calculated the reconstruction error of the pixel intensity value I at the location (x, y) in frame t as follows:

$$p(x, y, t) = \|I(x, y, t) - f(I(x, y, t))\|_2, \tag{1}$$

where f represents our two-stream model. We calculate the Euclidean distance between the initial pixel of the t-th frame and the pixel of the reconstructed frame as the reconstruction error of the pixel. For each frame, we compute the reconstruction error probability by summing up all the pixel-based probabilities.

$$R(t) = \sum_{(x,y)} p(x, y, t). \tag{2}$$

After calculating the reconstruction error of the spatial model $R_S(t)$ and temporal model $R_T(t)$, respectively. We calculated the reconstruction error of the fusion model $R_F(t)$. Due to the different dimensions of the two models. The fused reconstruction error $R_F(t)$ can be obtained using the following equation:

$$R_F(t) = R_S(t) * R_T(t). \tag{3}$$

After we define the reconstruction error probability of a frame as $R_F(t)$, the abnormality score can be defined as follows:

$$S_a(t) = \frac{R_F(t) - \min_t R_F(t)}{\max_t R_F(t) - \min_t R_F(t)}. \tag{4}$$

The abnormality score $S_a(t)$ corresponds to the level of abnormality of each frame in the video, which plays a role in indicating the confidence of detection results. On the other hand, the regularity score $S_a(t)$ corresponds to the level of normality can be defined as follows:

$$\mathrm{s}_r(t) = 1 - s_a(t). \tag{5}$$

Assume that the regularity score of the current frame is relatively low. In this case, the possibility of abnormality in the video frame is high. On the other hand, if there is no abnormality in the video frame, the regularity score of the frame should also be relatively high.

3.2. Feature Map Visualization Algorithm. This algorithm is based on the visualization of the feature map in CNNs, transforming the image's interior features into a visible and understandable image pattern, which helps us clearly understand the features learned by the model. The feature learning of convolutional neural networks is an incomprehensible process for humans to confirm whether it has learned features that have a natural effect on prediction, for example, in detecting abnormal events, whether it judges or predicts an abnormality based on understanding the abnormal behaviour characteristics of the monitored object. After extracting each convolutional layer's features, the model generated a certain number of feature maps. We can use the visual feature map to explain the features learned by the model in each convolutional layer, helping us understand and trust the final result. Inspired by Grad-CAM [26], we combined the feature map of the convolutional layer with the input image to generate the heatmap and display the features learned by the model's convolutional layer in the form of objects in the heatmap. Compared with other studies, the main difference in our work performs the reconstructed video sequence that does not need to get the gradient of the convolutional layers. Therefore, we directly superimposed the feature map into the abnormal frame in the form of a heatmap without changing its gradient. The method proposed in this paper can realize visual feature maps in any convolutional layer in spatial and temporal models and has particular applicability in deep learning models in other fields.

Assuming that the current convolutional layer of the model has n feature maps, the convolutional layer here can be any layer, denoted as $A^1, A^2, \ldots, A^n$, the size of the feature map is $r * c$, $S = r * c$.The pixel values of the k-th row and the j-th column of the i-th feature map are A^i_{kj}. The activation mapping LAM for this layer is as follows:

$$\begin{aligned} LAM &= \mathrm{ReLU}\left(\sum_{i=1}^{n} w_i * A^i\right), \\ A^i &= \frac{1}{S}\sum_{k=1}^{r}\sum_{j=1}^{c} A^i_{kj}, \\ LAM &= \mathrm{ReLU}\left(\frac{1}{S}\sum_{i=1}^{n}\sum_{k=1}^{r}\sum_{j=1}^{c} w_i * A^i_{kj}\right). \end{aligned} \tag{6}$$

Among them, w_i corresponding to the weight of each neuron, the value is 1 in this paper. Because the spatial model is based on the Autoencoder, it learns the characteristics of many monitored objects in the video frame. Unlike the object classification task, the video anomaly detection task needs to learn the features of all objects. Therefore, each feature map may contain multiple object parts, adding each feature map in a one-to-one ratio. Using the ReLU function is that only the part with the feature value greater than 0 is needed, that is, the part that the model focuses on.

3.3. Abnormal Detection. This method combines the feature map visualization algorithm with the two-stream network to solve the low reliability of the deep learning model in video anomaly detection. When abnormal events were detected, the internal logic of the model was explained through a heat map. Thus, the method was divided into two parts: (1) anomaly detection based on a two-stream network; (2) visualization based on the heat map. The two parts can operate independently or put together.

The anomaly detection process can also be divided into two parts (Figure 4): (1) merge the spatial flow network and the temporal flow network to detect anomalies; (2) use a separate subnetwork to learn features to detect anomalies. Both parts need to preprocess the video, decompose the video clip into video frames, learn the features, reconstruct the video frame, and calculate the frame's regularity score. Temporal and spatial networks are autoencoders, and they can generate reconstructed frames through reconstruction methods and calculate regularity scores. Because the inputs of the two networks are different, the spatial model was input in a single video frame, which reduced the memory consumption of model training. The temporal model needs to model the correlation between adjacent video frames, so the Conv-LSTM layer was used and input in four video frames. The advantage of the model design was that relatively few training samples were used to model spatial features, and learned spatial and temporal features were separated through two submodels. Then, the two models can be fused to achieve better anomaly detection results. In addition, the model design is relatively simple, which improves the speed of learning and finding anomalies. Through model fusion, the detection performance can be guaranteed within a reasonable range.

4. Experiment and Results

4.1. Datasets. We conducted experiments on four public benchmark datasets: Avenue [47], UCSD, Ped2 [48], and Subway Exit and Entrance datasets [49]. The Avenue dataset has 16 training video clips and 21 test video clips. The duration of each clip varies from less than one minute to two minutes. The UCSD Ped2 dataset is where pedestrians move parallel to the camera plane, containing 16 training and 12 test videos. The Subway Entrance video is 1 hour and 36 min long and consists of 66 abnormal events, while the Subway Exit dataset includes 19 abnormal events, and the duration is 43 minutes. Since the subway video clips are too long and the amount of data is too large, we only used the first 5 minutes of the Subway Exit video for training and the first 15 minutes of the Subway Entrance video. Then, the test dataset is divided into 4 and 6 test videos. Each test video is a continuous segment, and the approximate duration is 10 minutes and 13 minutes, respectively. (The former is the Subway Exit dataset; the latter is the Subway Entrance dataset.) Table 1 shows the details of the datasets.

4.2. Model Configuration. Here, we provide a detailed configuration of our method in Table 2. Moreover, all experiments running on a PC equipped with a GeForce RTX2080 GPU, 64G RAM, and running the Windows 10 operating system.

4.3. Experiment on Anomaly Detection

4.3.1. Quantitative Analysis: Frame-Level AUC. To better compare with other methods, all the experiments are carried out on the same PC with Intel CPU I7 8700K, NVIDIA GTX 2080, and 64G RAM. If a frame contains at least one abnormal event, it is considered as a correct detection. This detection is compared to the frame-level ground-truth label. The area under the curve (AUC) and the equal error rate (EER) are the evaluation's two metrics. Furthermore, some contemporary documents [9, 10] believe that the EER evaluation criteria are a severe sample imbalance between normal and abnormal events. Using EER as an indicator will be misleading in practical applications. We agree with this view and use AUC for evaluation, assuming that the local minimum within 50 frames belongs to the same abnormal event.

(1) Effectiveness Analysis. Table 3 presents the AUC of our method and a series of state-of-the-art methods [7, 10–12, 14, 18, 20] on the Avenue, the UCSD Ped2, and the Subway Entrance and Exit datasets. As expected, our model performs the best performance on the avenue and subway entrance and exit datasets. In addition, although the version in Avenue and Ped datasets appears to be slightly lower than that in the other complicated architectures, it is still significantly higher than that of lightweight models and that single-level models. These results indicate that a multilevel model [14] or 3D indicator [12] can perform better in crowd-scene, such as the UCSD Ped2 dataset. However, the time cost of these methods was also higher. Besides, comparing our spatial model and temporal model and the fusion model, temporal and spatial model have their advantages and disadvantages. Still, the fusion model performs better than the former two on all data sets.

(2) Time-Cost Analysis. Besides the effectiveness analysis, we also compare the computation time cost of the proposed approach. Since the proposed methods are based on the reconstruction techniques with deep learning, the model during the test is compared with other reconstruction-based deep methods. Table 4 shows the average computation time of different deep ways. Only four video frames in the temporal stream and a single video frame in the spatial stream generate the reconstruction error in our process. Thus, less time is needed for reconstruction error computation. The result shows the proposed approach is comparable with other methods.

4.3.2. Qualitative Analysis: Visualizing Frame Regularity. The regularity score graphs obtained by the spatial and temporal models are similar, so only the spatial model's regularity scores are shown. Figures 5–8 illustrate the regularity score of each frame on the Avenue, UCSD Ped2, Subway Entrance, and Exit video, respectively. When an anomaly is detected, the regularity score of the anomaly frame is significantly decreased. Further, our model can also detect unlabelled abnormal events.

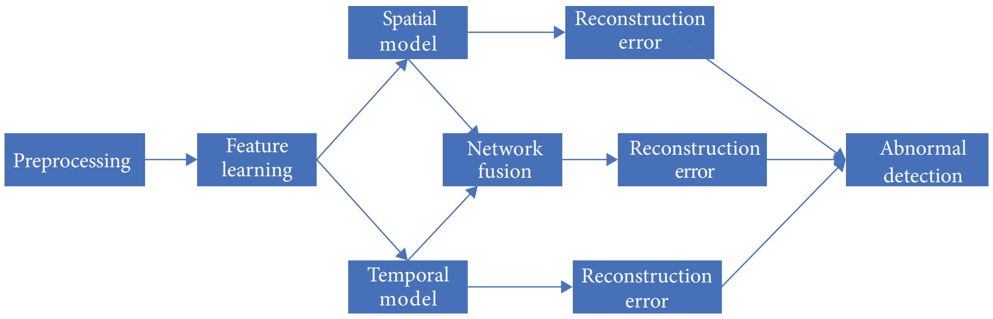


Figure 4: Anomaly detection flowchart. The first step is to process the video, learn the features through the two-stream network, and calculate the reconstruction error to detect anomalies.

Table 1: Details of the datasets.

Dataset	Frames	Training frames	Testing frames
Subway	125475	22500	102975
UCSD Ped	18560	9350	9210
CUHK Avenue	30652	15328	15324

Table 2: The parameter settings of our method.

Parameter	Value
Height	160
Width	240
Batch size	16
Lr	0.01
Epoch	200
Optimizer	SGD
Stride	4
Loss	MSE

Table 3: Comparison of area under ROC curve (frame-level AUC) of different methods.

Method	Avenue	Ped2	Subway Entrance	Subway Exit
ST-AE [20]	76.5	81.7	81.8	86.4
Unmasking [10]	80.6	82.2	70.6	85.7
GMFC-VAE [11]	78.6	84.9	83.7	87.4
RBM [12]	78.7	86.4	—	—
DAEs + cGAN [14]	73.6	86.1	84.1	87.3
ConvAe [7]	74.3	79.7	84.9	83.9
Conv-LSTM-AE [18]	76.4	82.9	83.3	86.4
Spatial model	78.1	83.8	84.7	90.2
Temporal model	77.8	84.0	85.0	85.4
Spatial + temporal	80.3	84.5	87.3	90.8

Higher AUC is better.

Table 4: Comparison of the average computation time (per epoch) on four data sets.

Method	Avenue (m)	Ped2 (m)	Subway Entrance (m)	Subway Exit (m)
ST-AE [20]	180	30	640	360
ConvAe [7]	244	40	320	120
Conv-LSTM-AE [18]	312	73	766	452
GMFC-VAE [11]	220	65	712	432
DAEs + cGAN [14]	430	194	904	642
Spatial model	19	3	24	8
Temporal model	95	15	120	50
Spatial + temporal	142	24	182	70

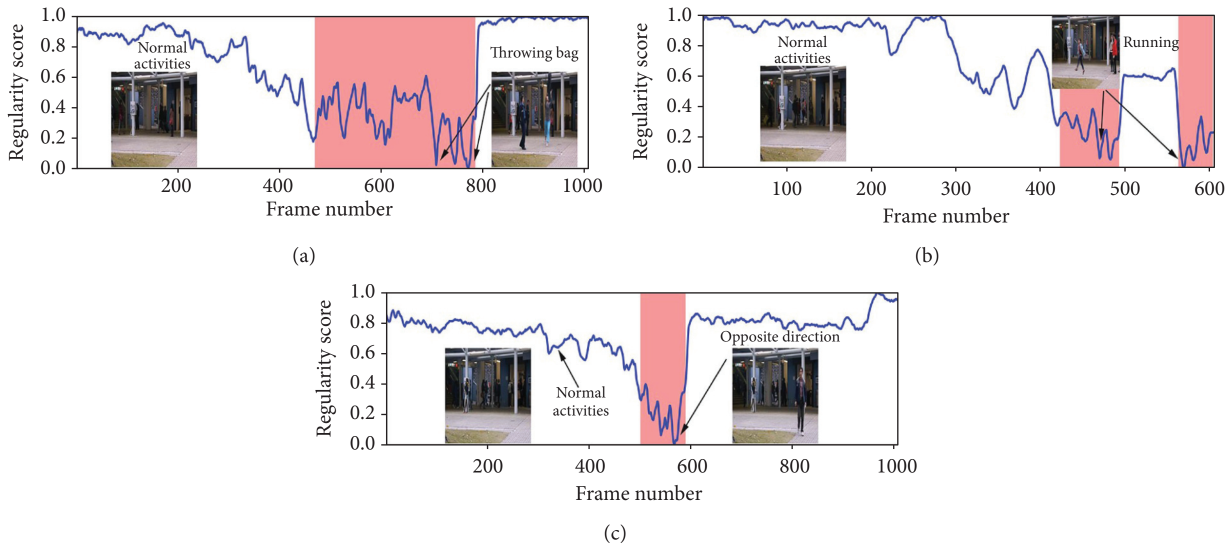


FIGURE 5: Regularity score of video #5, #7, and #15 from the Avenue video.

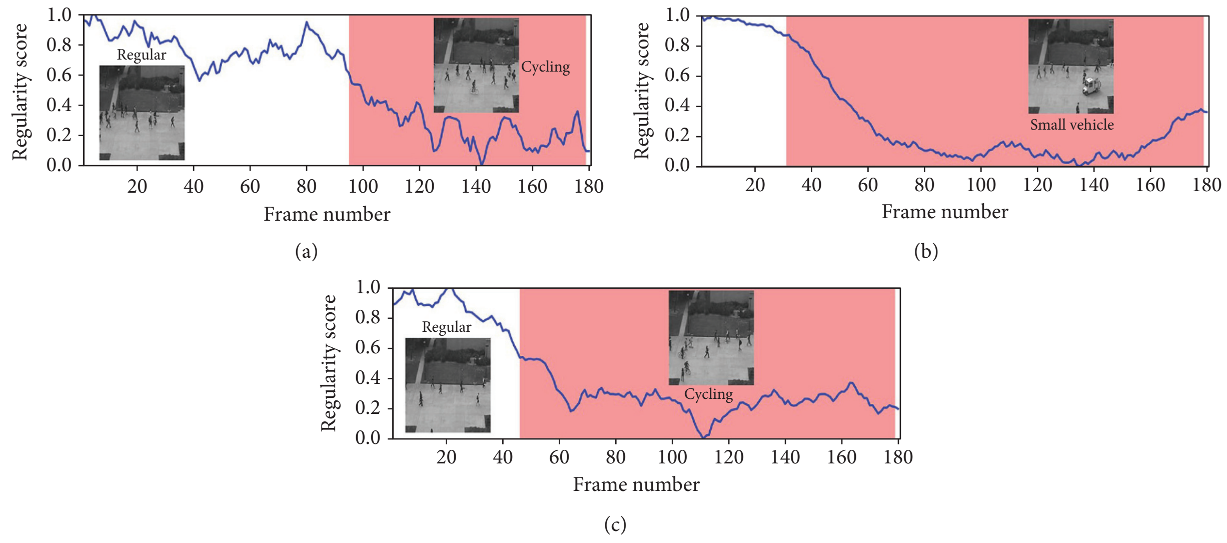


FIGURE 6: Regularity score of videos #2, #4, and #7 from the UCSD Ped2 video.

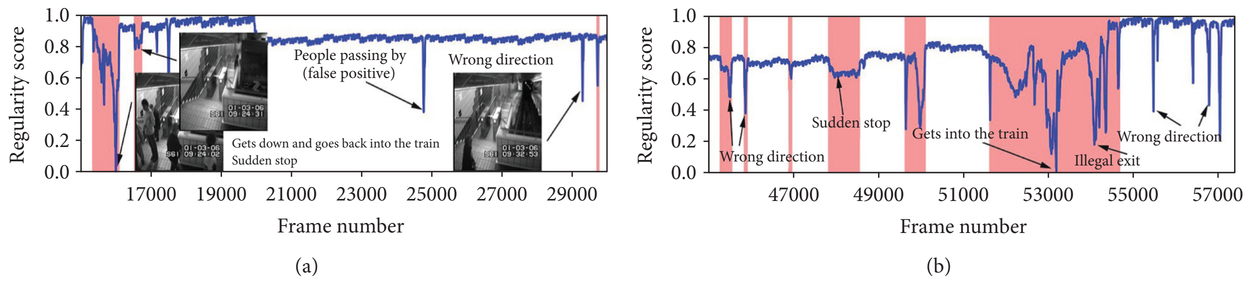
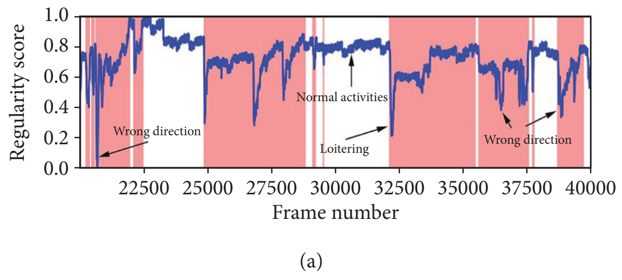


FIGURE 7: Regularity scores of frames 15000–30000 and 45000–57500 from the Subway Exit dataset.

Although our training process only used the usual scenes in the data set, our method can detect abnormal events that do not appear in the ordinary scene (Figure 5). For example, people enter the subway station from the subway station's exit for some prominent abnormal events and then enter and take the subway from here. As can be seen from the figures, the detection results match well with the ground-truth frames. The lower regularity scores correspond to abnormal



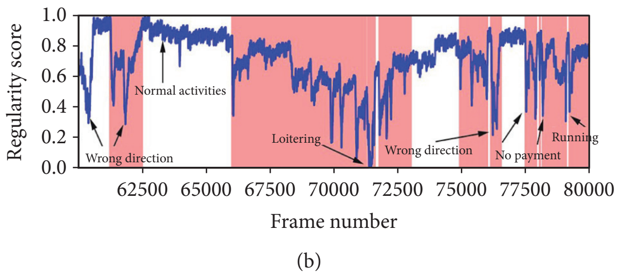


Figure 8: Regularity score of frames 20000–40000 and 80000–100000 from the Subway Entrance dataset.

events, while high regularity scores correspond to regular video frames. In the regularity score graph, the blue line is the regularity score of the video frame, and the red part is the abnormal event occurrence area marked by the ground truth. In Figures 5–7, according to the ground-truth anomaly labels, we can easily find abnormal events by setting the threshold to 0.5, excluding any false-positive detection. However, in some scenarios, false-positive detection will occur when the threshold is set to 0.5.

To better analyze the performance of our method, we also plot the anomaly score from Avenue, Ped, and Subway datasets. Figures 9–12 provide the detected events and the corresponding anomaly scores on the related data sets, with the anomaly score curve of the spatial, temporal, and two-stream fusion. The peak color regions indicate the frame-level ground-truth label of abnormal events. As can be seen from the figures, the detection results match well with the ground-truth frames.

4.4. Post Hoc Interpretability with Feature Visualization. Besides the quantitative and qualitative analysis, we use a heatmap to visualize the features of abnormal behaviour, such as skateboarding on the sidewalk, or entering the subway without playing, etc. Since the first three layers of the model's learned features are similar, this paper only shows the visualized heatmap of the first convolutional layer.

Figures 13–16 provide different visualization of the same data and show the features of the first convolutional layer on the Avenue dataset, UCSD Ped2 dataset, Subway Entrance, and Exit scenes, respectively. The frames are containing abnormal events and ordinary events. We achieve comparable results with the other two leading methods, and comparison experiments show that our method can detect anomalous objects well. Figure 13 shows our model learns a specific behaviour characteristic of people, such as losing the packet or walking around. As the running person is too fast, detecting abnormal behaviour characteristics is not so obvious. Figure 14 shows that our model is more interested in pedestrians walking and riding bicycles or carts. As shown in Figures 15 and 16, our model is interested in the people and characteristics of the track or train. These features can help our model identify the subway entrance and exit scene and the two videos' anomalies. In contrast, Grad-CAM [26] only visualizes some abnormal object regions, and Grad-CAM [27] visualizes many abnormal object regions.

Figures 13–16 show that our model can learn visual appearances and motion in the scene, helping to understand images and infer abnormal events. For example, Figure 13 shows an example illustrating the appearance and contour of vehicles with a darker color. Similarly, Figure 16 shows a person jumping over the fence and entering the subway exit. Thus, the model learns that the visual impressions and behaviours of individuals, combined with the scene. Therefore, it can be concluded that an abnormal event has occurred here. Therefore, our feature map visualization experiment can also verify the accuracy and authenticity of the abnormal detection results. Besides, the visualization method can explain the learning process of the model, but the visualization result will not affect the learning process and the final detection. Therefore, the interpretation method of this article can be considered as post hoc interpretability.

Conceptually, abnormal events are emerging from uncommon objects. Thus, while visualization of a trained model provides insight into its operation, it can also assist with selecting anomalous objects in abnormal video frames. The critical question is if the model identifies the object's location in the image with unnatural object detection approaches.

4.5. Discussion. Our results suggest that the proposed method enables fast and reliable detection of abnormal events, with label-free identification of abnormal events. In the quantitative experiment introduced in 4.3.1, we used four video frames in the temporal stream and a single video frame in the spatial stream to generate the reconstruction error, while number frames preferring ten were most frequently used by other methods. Using only the spatial or temporal stream cannot cause the best result. However, with the information from the two-stream fused, the model has improved efficiency compared with a single stream, while the accuracy is also competitive. It should also be noted that our current model is lightweight and does not consider the complete appearance and motion of the video scenario. Therefore, the training process in our method does not reconstruct all the changes in the properties of appearance and motion, and it may be weak compared with other techniques in a particular dataset—for example, GMFC-VAE in Ped2.

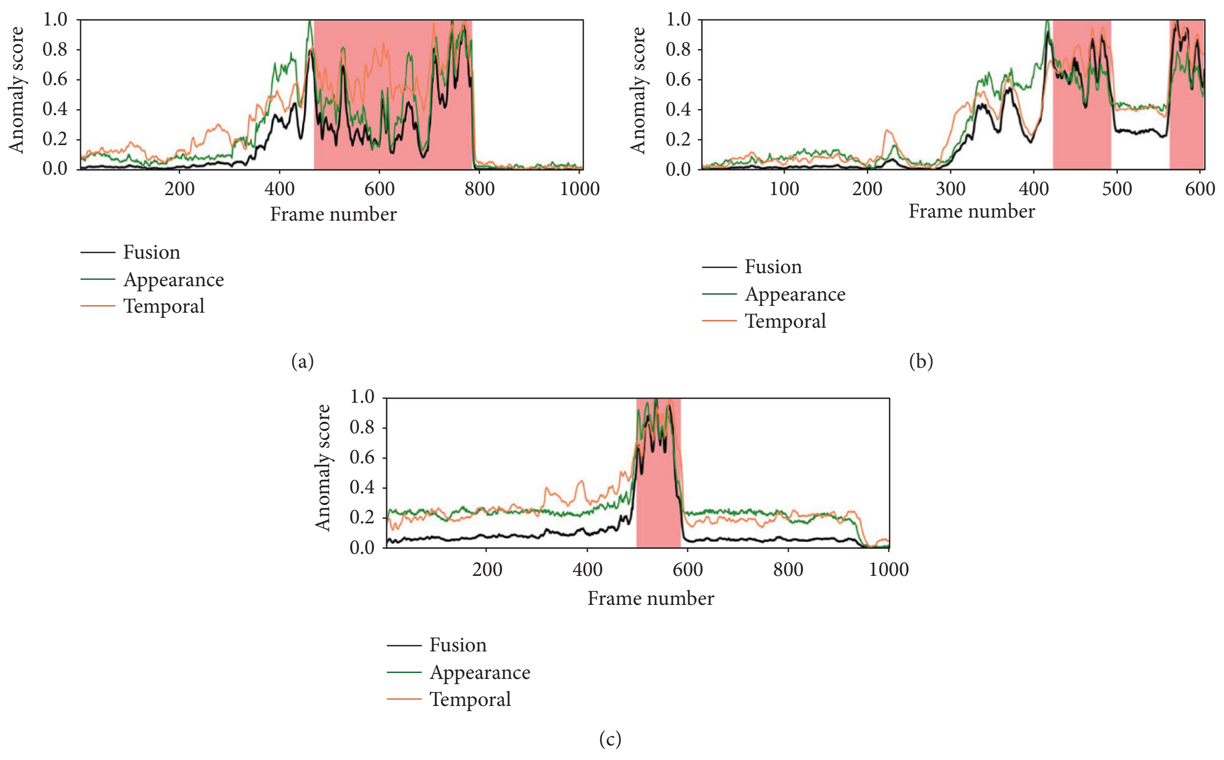


FIGURE 9: Anomaly score of videos #5, #7, and #15 from the Avenue video.

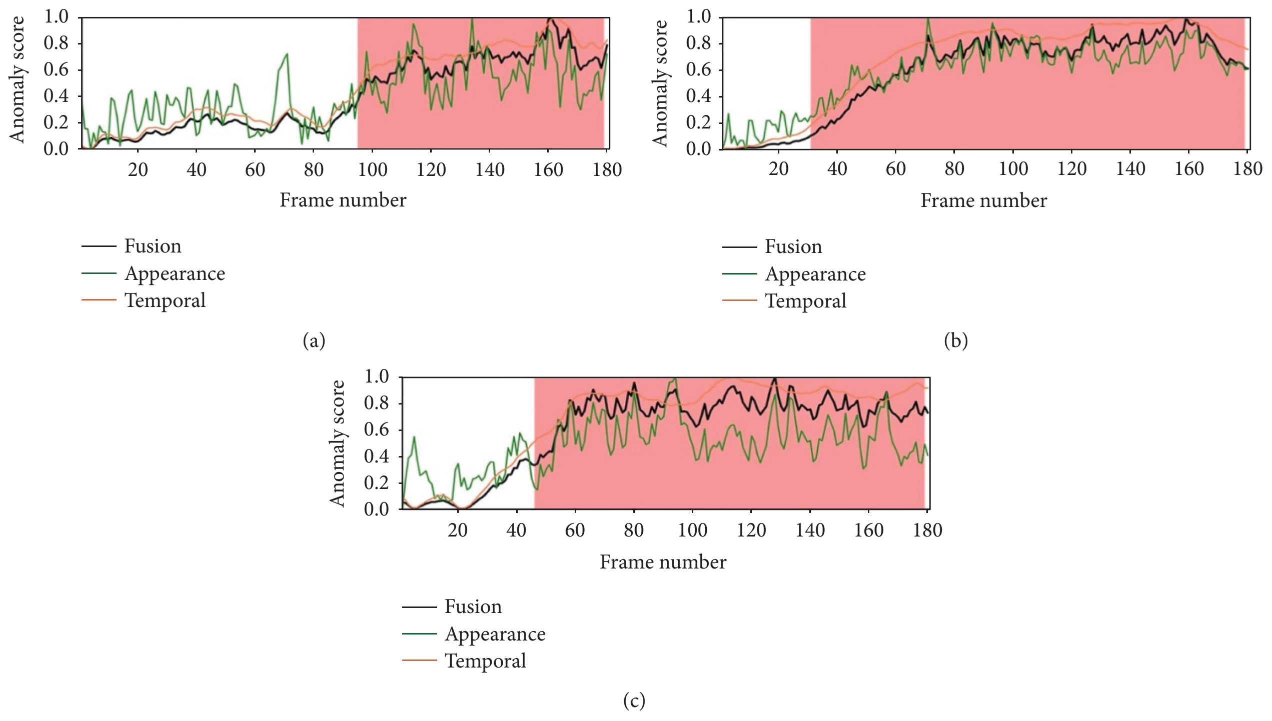
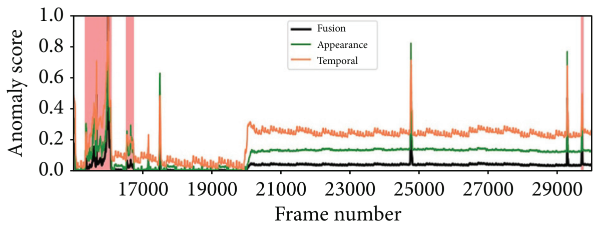
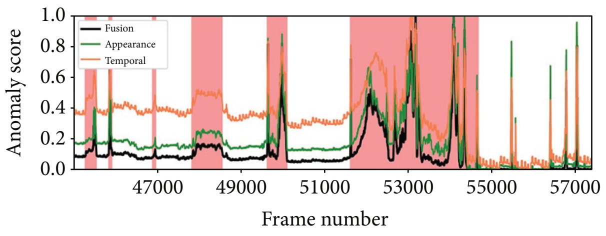


FIGURE 10: Anomaly score of videos #2, #4, and #7 from the UCSD Ped2 video.

Compared with other anomaly detection methods [5, 6, 16–20], we use a heat map to visualize the internal logic of the video frames, which is more interested in the darker part. Therefore, we can better understand the network's learning process and the basis for making abnormal behaviour judgments. We also compare the visualization of feature maps of different convolutional layers. Due to the relatively small number of layers, we find the features

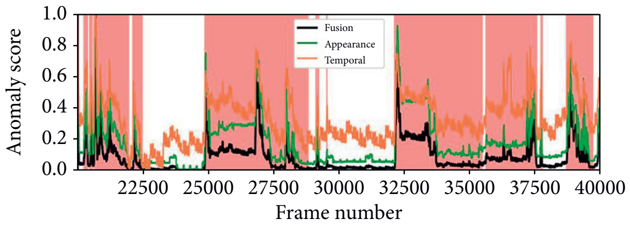


(a)

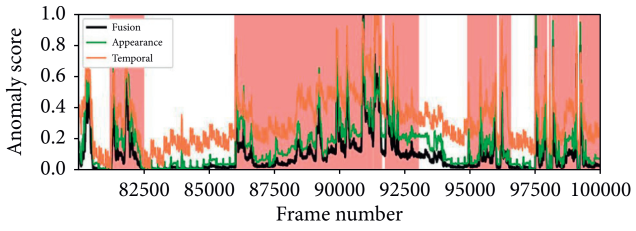


(b)

Figure 11: Anomaly scores of frames 15000–30000 and 45000–57500 from the Subway Exit dataset.



(a)



(b)

Figure 12: Anomaly score of frames 20000–40000 and 80000–100000 from the Subway Entrance dataset.

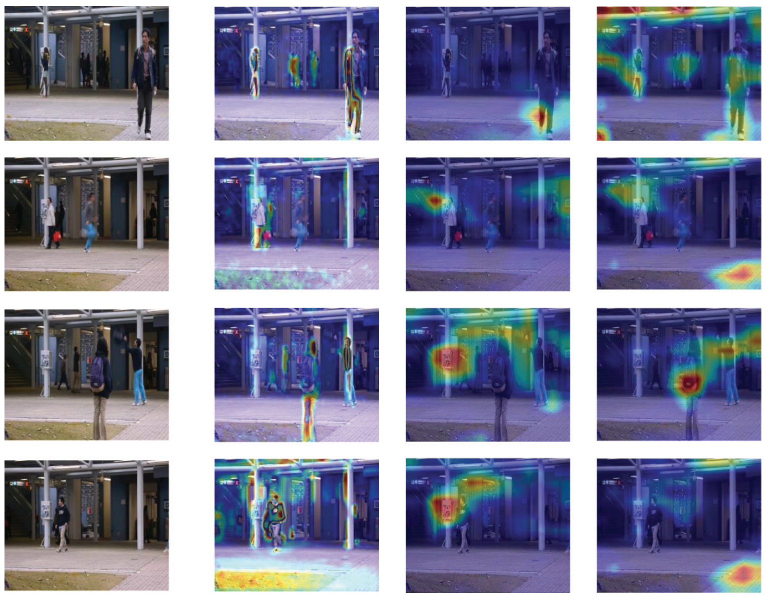


Figure 13: Feature visualization results on our method, Grad-CAM, and Score-CAM on Avenue dataset. Top-3 rows are abnormal video frames, and the last row is a normal video frame. (a) Initial video frame. (b) Our method. (c) Grad-CAM. (d) Score-CAM.

learned by the first and second convolutional layers are almost the same. They are relatively low-level edge and contour information rather than high-level abstract details. Our method exhibited interpretability and much better location stability than other anomaly detection methods.

In summary, these results highlight the effectiveness and high efficiency of the proposed method in abnormal event detection. However, although it can show interpretability in abnormal event detection, it is challenging to present interpretable loss terms in end-to-end training.

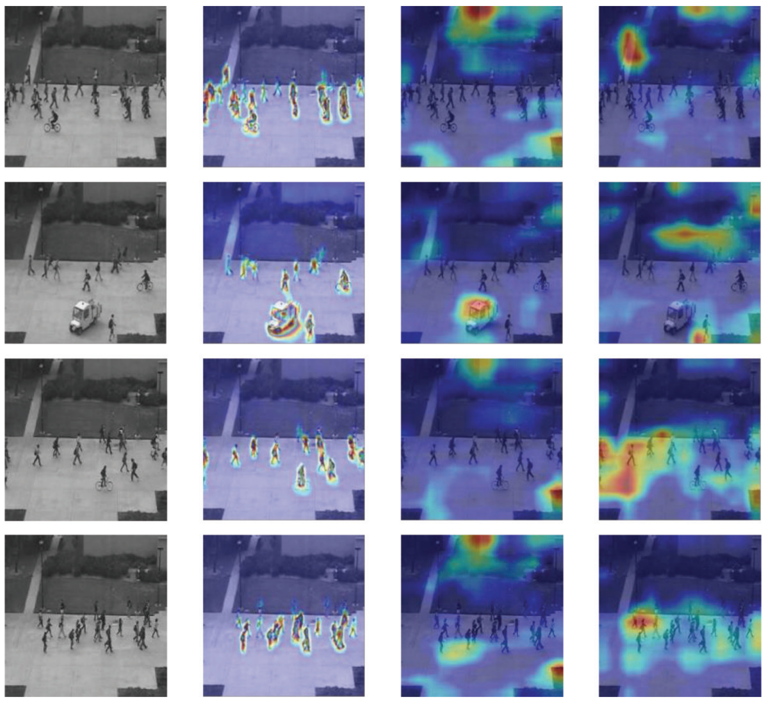


Figure 14: Feature visualization results on our method, Grad-CAM, and Score-CAM on UCSD Ped2 dataset. Top-3 rows are abnormal video frames, and the last row is a normal video frame. (a) Initial video frame. (b) Our method. (c) Grad-CAM. (d) Score-CAM.

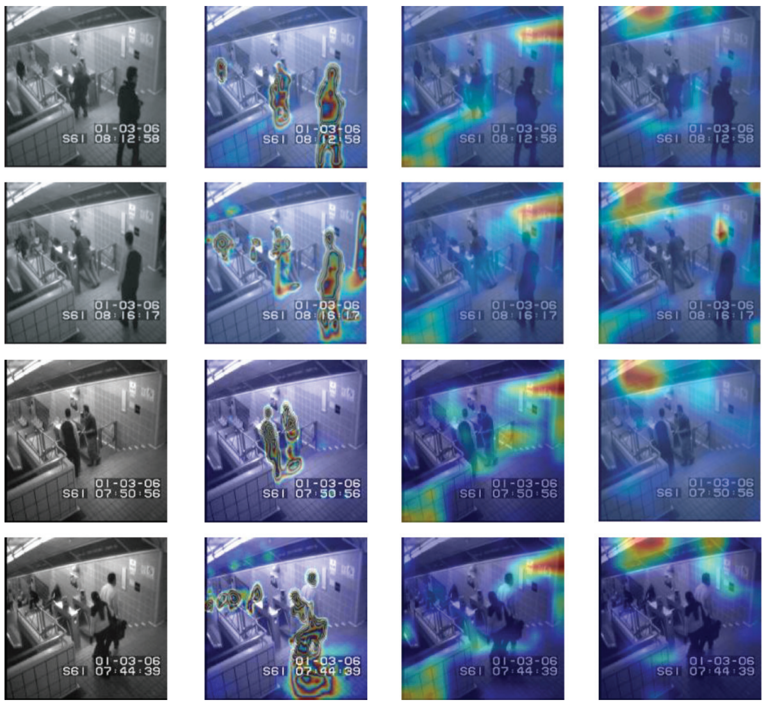


Figure 15: Feature visualization results on our method, Grad-CAM, and Score-CAM on Subway Entrance dataset. Top-3 rows are abnormal video frames, and the last row is a normal video frame. (a) Initial video frame. (b) Our method. (c) Grad-CAM. (d) Score-CAM.

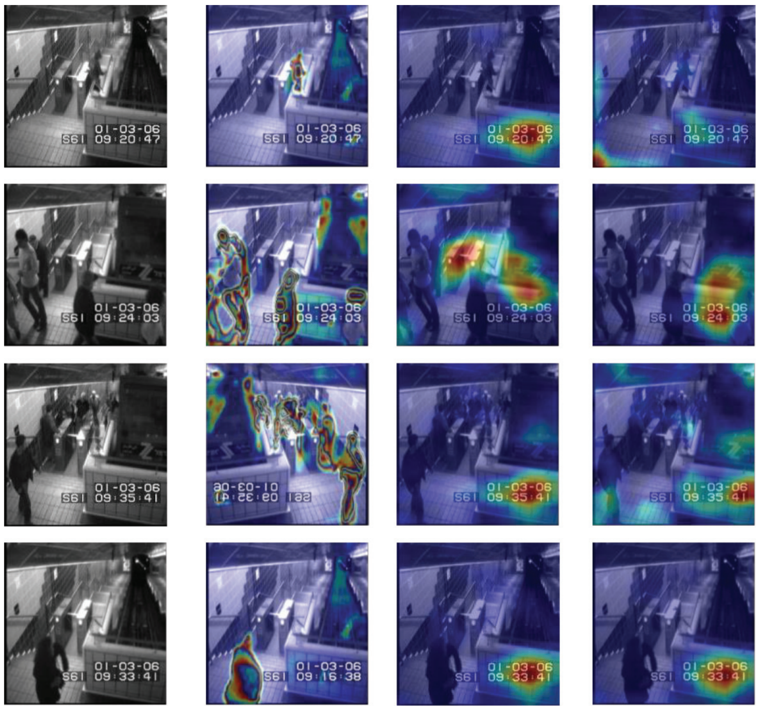


Figure 16: Feature visualization results on our method, Grad-CAM, and Score-CAM on Subway Exit dataset. Top-3 rows are abnormal video frames, and the last row is a normal frame. (a) Initial video frame. (b) Our method. (c) Grad-CAM. (d) Score-CAM.

5. Conclusions and Future Work

We have presented a prevailing method to detect abnormal events from videos to intensify detection ability and feature interpretability with a two-stream framework. Our approach fuses the visual appearances, behavioural characteristics, and motion of the video object and can determine abnormal events from many regular activities. To critically assess the robustness of detecting in capturing abnormal events, we performed several challenging data sets that allow our algorithm to operate robustly for long periods in various scenes, including crowded ones. Experiments have shown that our method is accurate and robust to noise. Furthermore, the visualization of feature maps semanticizes the internal logic.

Meanwhile, applying explainable deep learning methods to anomaly detection will be a future research direction. It has excellent benefits for handling abnormal events and even preventing abnormal events from happening in advance, which has great significance in public security.

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 no conflicts of interest.

Acknowledgments

This work was supported by the National Nature Science Foundation of China (41971365) and the Chongqing Research Program of Basic Science and Frontier Technology (cstc2019jcyj-msxmX0131).

References

[1] S. Yan, J. S. Smith, W. Lu et al., "Abnormal event detection from videos using a two-stream recurrent variational autoencoder," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 12, no. 1, pp. 30–42, 2018.

[2] Y. Yuan, Y. Feng, and X. Lu, "Statistical hypothesis detector for abnormal event detection in crowded scenes," *IEEE Transactions on Cybernetics*, vol. 47, no. 11, pp. 3597–3608, 2016.

[3] C. Feichtenhofer, H. Fan, J. Malik et al., "Slowfast networks for video recognition," in *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6202–6211, IEEE, Seoul, Republic of Korea, October 2019.

[4] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1933–1941, IEEE, Las Vegas, NV, USA, June 2016.

[5] D. Xu, E. Ricci, Y. Yan et al., "Learning deep representations of appearance and motion for anomalous event detection," 2015, https://arxiv.org/abs/1510.01553.

[6] D. Xu, Y. Yan, E. Ricci, and N. Sebe, "Detecting anomalous events in videos by learning deep representations of appearance and motion," *Computer Vision and Image Understanding*, vol. 156, pp. 117–127, 2017.

[7] M. Hasan, J. Choi, J. Neumann, A. K. Roy-Chowdhury, and L. S. Davis, "Learning temporal regularity in video sequences," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 733–742, IEEE, Las Vegas, NV, USA, June 2016.

[8] R. T. Tonescu, S. Smeureanu, M. Popescu et al., “Detecting abnormal events in video using narrowed normality clusters,” in *Proceedings of the 2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1951–1960, IEEE, Waikoloa, HI, USA, January 2019.

[9] A. Del Giorno, J. A. Bagnell, and M. Hebert, “A discriminative framework for anomaly detection in large videos,” in *Proceedings of the European Conference on Computer Vision*, pp. 334–349, ECCV, Amsterdam, Netherlands, October 2016.

[10] R. Tudor Ionescu, S. Smeureanu, B. Alexe et al., “Unmasking the abnormal events in video,” in *Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2895–2903, IEEE, Venice, Italy, October 2017.

[11] Y. Fan, G. Wen, D. Li et al., “Video anomaly detection and localization via Gaussian mixture fully convolutional variational AutoEncoder,” *Computer Vision and Image Understanding*, vol. 195, Article ID 102920, 2020.

[12] H. Vu, T. D. Nguyen, A. Travers, S. Venkatesh, and D. Phung, “Energy-based localized anomaly detection in video surveillance,” in *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, pp. 641–653, Jeju, Republic of Korea, May 2017.

[13] M. Sabokrou, M. Fayyaz, M. Fathy, and R. Klette, “Deep-cascade: Cascading 3d deep neural networks for fast anomaly detection and localization in crowded scenes,” *IEEE Transactions on Image Processing*, vol. 26, no. 4, pp. 1992–2004, 2017.

[14] H. Vu, T. D. Nguyen, T. Le, W. Luo, and D. Phung, “Robust anomaly detection in videos using multi-level representations,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 5216–5223, AAAI Press, Palo Alto, CA, USA, January 2019.

[15] W. Liu, W. Luo, D. Lian et al., “Future frame prediction for anomaly detection–a new baseline,” in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6536–6545, IEEE, Salt Lake City, UT, USA, June 2018.

[16] W. Luo, W. Liu, and S. Gao, “A revisit of sparse coding based anomaly detection in stacked rnn framework,” in *Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 341–349, IEEE, Venice, Italy, October 2017.

[17] M. Ravanbakhsh, M. Nabi, H. Mousavi et al., “Plug-and-play cnn for crowd motion analysis: An application in abnormal event detection,” in *Proceedings of the 2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1689–1698, IEEE, Lake Tahoe, NV, USA, March 2018.

[18] W. Luo, W. Liu, and S. Gao, “Remembering history with convolutional lstm for anomaly detection,” in *Proceedings of the 2017 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 439–444, IEEE, Hong Kong, China, July 2017.

[19] S. Smeureanu, R. T. Ionescu, M. Popescu, and B. Alexe, “Deep appearance features for abnormal behaviour detection in video,” in *Proceedings of the International Conference on Image Analysis and Processing*, pp. 779–789, ICIAP, Catania, Italy, September 2017.

[20] Y. S. Chong and Y. H. Tay, “Abnormal event detection in videos using spatiotemporal Autoencoder,” in *Proceedings of the International Symposium on Neural Networks*, pp. 189–196, ISNN, Hokkaido, Japan, June 2017.

[21] X. Guo, B. Hou, B. Ren, Z. Ren, and L. Jiao, “Network pruning for remote sensing images classification based on interpretable CNNs,” *IEEE Transactions on Geoscience and Remote Sensing*, pp. 1–15, 2021.

[22] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” 2013, https://arxiv.org/abs/1312.6114.

[23] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *Proceedings of the European conference on computer vision*, pp. 818–833, ECCV, Zurich, Switzerland, August 2014.

[24] A. Mahendran and A. Vedaldi, “Understanding deep image representations by inverting them,” in *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5188–5196, IEEE, Boston, MA, USA, June 2015.

[25] J. T. Springenberg, A. Dosovitskiy, T. Brox et al., “Striving for simplicity: The all convolutional net,” 2014, https://arxiv.org/abs/1412.6806.

[26] R. R. Selvaraju, M. Cogswell, A. Das et al., “Grad-cam: Visual explanations from deep networks via gradient-based localization,” in *Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 618–626, IEEE, Venice, Italy, October 2017.

[27] H. Wang, Z. Wang, M. Du et al., “Score-CAM: score-weighted visual explanations for convolutional neural networks,” in *Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 24-25, IEEE, Seattle, WA, USA, June 2020.

[28] A. Dosovitskiy and T. Brox, “Inverting visual representations with convolutional networks,” in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4829–4837, IEEE, Las Vegas, NV, USA, June 2016.

[29] B. Zhou, A. Khosla, A. Lapedriza et al., “Object detectors emerge in deep scene cnns,” 2014, https://arxiv.org/abs/1412.6856.

[30] Q. Zhang, R. Cao, F. Shi et al., “Interpreting cnn knowledge via an explanatory graph,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 4454–4463, AAAI Press, New Orleans, LA, USA, Febuary 2018.

[31] Q. Zhang, R. Cao, Y. N. Wu et al., “Growing interpretable part graphs on convnets via multi-shot learning,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 2898–2906, AAAI Press, San Francisco, CA, USA, Febuary 2017.

[32] Q. Zhang, Y. Yang, H. Ma et al., “Interpreting cnns via decision trees,” in *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6261–6270, IEEE, Long Beach, CA, USA, June 2019.

[33] Q. Zhang, Y. N. Wu, and S. C. Zhu, “Interpretable convolutional neural networks,” in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8827–8836, IEEE, Salt Lake City, UT, USA, June 2018.

[34] T. Wu, X. Li, X. Song et al., “Interpretable r-cnn,” 2017, https://arxiv.org/abs/1711.05226.

[35] Y. Jeon, A. Watanabe, S. Hagiwara et al., “Interpretable and lightweight 3-D deep learning model for automated ACL diagnosis,” *IEEE Journal of Biomedical and Health Informatics*, vol. 99, 2021.

[36] T. Pianpanit, S. Lolak, P. Sawangjai, T. Sudhawiyangkul, and T. Wilaiprasitporn, “Parkinson’s disease recognition using SPECT image and interpretable AI: A tutorial,” *IEEE Sensors Journal*, 2021.

[37] S. Akcay, A. Atapour-Abarghouei, and T. P. Breckon, “Ganomaly: semi-supervised anomaly detection via adversarial training,” in *Proceedings of the Asian Conference on Computer Vision*, pp. 622–637, Perth, Australia, December 2018.

[38] Y.-H. Kwon and M.-G. Park, “Predicting future frames using retrospective cycle GAN,” in *Proceedings of the CVPR 2019:*

IEEE Conference on Computer Vision and Pattern Recognition, pp. 1811–1820, Long Beach, CA, USA, June 2019.

[39] T. Schlegl, P. Seeböck, S. M. Waldstein, G. Langs, and U. Schmidt-Erfurth, "F-AnoGAN: fast unsupervised anomaly detection with generative adversarial networks," *Medical Image Analysis*, vol. 54, pp. 30–44, 2019.

[40] B. R. Kiran, D. M. Thomas, and R. Parakkal, "An overview of deep learning based methods for unsupervised and semi-supervised anomaly detection in videos," *Journal of Imaging*, vol. 4, no. 2, pp. 1–15, 2018.

[41] W. Liu, W. Luo, D. Lian, and S. Gao, "Future frame prediction for anomaly detection—A new baseline," in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 6536–6545, Salt Lake City, UT, USA, June 2018.

[42] Y. Qiang, S. Fei, and Y. Jiao, "Anomaly detection based on latent feature training in surveillance scenarios," *IEEE Access*, vol. 9, pp. 68108–68117, 2021.

[43] J. Yu, Y. Lee, K. C. Yow, M. Jeon, and W. Pedrycz, "Abnormal event detection and localization via Adversarial event prediction," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 99, pp. 1–15, 2021.

[44] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," 2014, https://arxiv.org/abs/1406.2199.

[45] X. Shi, Z. Chen, H. Wang et al., "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," 2015, https://arxiv.org/abs/1506.04214.

[46] V. Patraucean, A. Handa, and R. Cipolla, "Spatio-temporal video autoencoder with differentiable memory," in *Proceedings of the International Conference on Learning Representations (2015)*, pp. 1–10, May 2016, https://arxiv.org/abs/1511.06309.

[47] C. Lu, J. Shi, and J. Jia, "Abnormal event detection at 150 fps in matlab," in *Proceedings of the 2013 IEEE International Conference on Computer Vision*, pp. 2720–2727, IEEE, Sydney, NSW, Australia, December 2013.

[48] V. Mahadevan, W. Li, V. Bhalodia, and N. Vasconcelos, "Anomaly detection in crowded scenes," in *Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1975–1981, IEEE, San Francisco, CA, USA, June 2010.

[49] A. Adam, E. Rivlin, I. Shimshoni, and D. Reinitz, "Robust real-time unusual event detection using multiple fixed-location monitors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 3, pp. 555–560, 2008.

Hindawi
Computational Intelligence and Neuroscience
Volume 2021, Article ID 9922697, 15 pages
https://doi.org/10.1155/2021/9922697

Research Article

Scaling Human-Object Interaction Recognition in the Video through Zero-Shot Learning

Vali Ollah Maraghi and Karim Faez

Department of Electrical Engineering, Amirkabir University of Technology (Tehran Polytechnic), Tehran, Iran

Correspondence should be addressed to Karim Faez; kfaez@aut.ac.ir

Received 11 March 2021; Revised 28 April 2021; Accepted 18 May 2021; Published 10 June 2021

Academic Editor: Miguel Cazorla

Recognition of human activities is an essential field in computer vision. The most human activity consists of the interaction between humans and objects. Many successful works have been done on human-object interaction (HOI) recognition and achieved acceptable results in recent years. Still, they are fully supervised and need to train labeled data for all HOIs. Due to the enormous space of human-object interactions, listing and providing the training data for all possible categories is costly and impractical. We propose an approach for scaling human-object interaction recognition in video data through the zero-shot learning technique to solve this problem. Our method recognizes a verb and an object from the video and makes an HOI class. Recognition of the verbs and objects instead of HOIs allows identifying a new combination of verbs and objects. So, a new HOI class can be identified, which is not seen by the recognizer system. We introduce a neural network architecture that can understand and represent the video data. The proposed system learns verbs and objects from available training data at the training phase and can identify the verb-object pairs in a video at test time. So, the system can identify the HOI class with different combinations of objects and verbs. Also, we propose to use lateral information for combining the verbs and the objects to make valid verb-object pairs. It helps to prevent the detection of rare and probably wrong HOIs. The lateral information comes from word embedding techniques. Furthermore, we propose a new feature aggregation method for aggregating extracted high-level features from video frames before feeding them to the classifier. We illustrate that this feature aggregation method is more effective for actions that include multiple subactions. We evaluated our system by recently introduced Charades challengeable dataset, which has lots of HOI categories in videos. We show that our proposed system can detect unseen HOI classes in addition to the acceptable recognition of seen types. Therefore, the number of classes identifiable by the system is greater than the number of classes used for training.

1. Introduction

Humans play a significant role in most of the activities that take place in the world. Human action recognition is one of the fundamental problems in computer vision and has many applications, such as video navigation, human-robot collaboration, and predicting human behavior for security purposes. Many human activities are made up of two parts: a verb and an object. The verb is what a man does on an object. In fact, a verb represents the movement of the human body. This type of activity is referred to as human-object interaction (HOI). For example, "opening the door" or "reading a book" has a verb and an object. Therefore, recognizing HOI is as important and challenging as recognizing human activities in the field of machine vision.

Many researchers have been working on human-object interaction (HOI) understanding [1–6]. HOI understanding can be followed in images or videos (sequence of frames). Distinguishing the full range of human activities in real environments is a significant challenge in computer vision. Some problems are as follows: large intraclass variation in actions, high variability in spatiotemporal scaling, human pose variations, occlusions, and, most importantly, the vast space of human activities. The most effective HOI recognition task methods are methods based on deep learning approaches that need a lot of labeled data [6–11]. The existing machine learning approaches for action understanding require fully annotated datasets. Some datasets have been prepared for this purpose, such as the "Humans Interacting with Common Objects" dataset for action classification (HICO) [3] and action detection (HICO-DET) [5] that are image benchmark. For HOI analysis in videos, not many datasets are provided. Charades dataset [12] was recently supplied for the HOI understanding tasks in video data. It has many human-object interaction categories and is suitable for action detection, HOI recognition, and video captioning purposes.

One of the essential HOI recognition challenges is the high number of possible categories in the real environment. Since the space of possible HOIs in real situations is enormous, list all possible HOI classes and obtain enough data for each group, and annotating the collected data is impractical. How can we reduce the need for training data for training an HOI recognizer model? Can a recognition model be taught with data from only part of the target classes? We focus on this question and tackle it through zero-shot learning for scaling HOI recognition in video data. In the recognition approaches based on zero-shot learning, the main class is decomposed into its components, and recognizing the main class components is applied instead of understanding the main category. So, the recognizer model learns the components of unseen classes that probably appear in other seen classes. In this case, if components of the novel HOI class appeared in other seen HOI classes, the model can recognize them and identify the new HOI class. Zero-shot learning for scaling HOI recognition is used previously for image data [13]. Our previous work presented a simple structure for zero-shot recognizing of HOI in video data [14].

In this work, we expand our previous work [14] and address the scaling of human-object interaction in video data through zero-shot learning. In this approach, the HOIs decompose into verbs and objects as the components of an HOI. For each input video containing an HOI, the detection system recognizes a verb and an object. For this purpose, the central proposed scheme has a two-branch deep neural network structure consisting of object recognition and verb recognition branches. A convolutional neural network (CNN) is used to extract the feature maps of each frame. We use recurrent neural networks (RNNs) in the verb recognition branch due to the video's temporal information. RNNs can represent the long dependencies in video data, which can help recognize verbs in the video.

We propose using lateral information to combine the verbs and the objects better to make valid verb-object pairs. It helps to prevent the detection of rare and probably wrong HOIs. The lateral information comes from word embedding techniques.

We also propose a new feature aggregation method for aggregating extracted high-level features from video frames before feeding them to the classifier. We use a local feature aggregation method that does not turn the entire extracted features space into a single space. We illustrate that this feature aggregation method is more effective for actions that include multi subactions.

We evaluate our proposed algorithm on the Charades dataset [12] and illustrate that our model can identify the novel HOI categories not seen by the model before. The Charades dataset has many human-object interaction categories and is suitable for action detection, HOI recognition, and video captioning purposes. This dataset contains 9848 video clips of HOIs captured in real environments. It has 157 categories of human activities, including some actions with "no interaction," 149, which can be considered valid verb-object pairs. This 149 category consists of 34 verbs and 37 objects. Also, we compare our model in a fully supervised manner with the best-reported methods on this dataset and show that our method's performance can be comparable to them.

This study's primary purpose is to reduce the need for data to train an HOI recognition system by increasing the number of identifiable HOIs without increasing HOIs in training data. We focus on this purpose through zero-shot learning, in which we decompose the HOI into a verb (human action) and an object and recognize them in the video. We use CNNs and RNNs for implementing our proposed algorithm.

In the rest of the paper, we review some related works in part 2. The model architecture and proposed algorithm are presented in Section 3. We present the experimental results, evaluations, and discuss the results in Section 4, and conclude in Section 5.

2. Related Works

2.1. Human Action and HOI Recognition. Initial works on understanding human activities were in modeling actions. Many works can be found that used semantics for modeling and understanding of activities [15]. The HOI modeling started with the affordances idea introduced by J. Gibson [16], and then some works were done in the field of functionality understanding of objects and verbs [17]. Several approaches have been used to model semantic relationships [18, 19] for HOI understanding. Modeling humans and objects' spatial relationships using the interactional features are introduced by Delaitre et al. [20]. Also, learning distributed representations of humans and objects by poselet [21] and phraselets [22] are proposed for HOI recognition. Most of these efforts require costly-labeled data (pose, body parts, and object segmentation, etc.), making it difficult to collect data for any type of activity and make them applicable for cases with a limited number of classes. In fact, they fail for cases with more classes.

Recently, with providing large datasets [3, 5, 23] and the success of neural network-based approaches in classification and recognition tasks, the problem of understanding and recognizing HOIs has received a lot of attention. Inspired by this impressive progress, the researchers tried to develop deep networks for video analysis applications such as action recognition [24–28] and HOI understanding [23, 29].

The recognition in video is more complicated than recognition in still images because of the complexity of video sequences' motion patterns. Therefore, the mere use of appearance cues for successful recognition may not be enough. Most existing approaches have introduced a two-stream framework that considers both temporal and spatial domains [25, 30–37]. Classifiers operate on the two streams of inputs, the RGB and the optical flow, as spatial and temporal cues, respectively. Motion cues are used separately from appearance cues for final representing in the video. Two streams are trained individually in the training phase, and the outputs of them fuse to predict the output class in the testing phase.

Some approaches use 3D networks that work by spatiotemporal convolutions [26]. These networks usually consider a short video interval with a predefined number of frames and encode the local and short-term motion patterns. For example, 15, 7, 16, and 2 frames are used in [26, 30, 38, 39]. Also, other types of spatiotemporal networks like RNNs [26] and the extended versions of them which are called Long Short-Term Memory (LSTM) [40] are used to describe the temporal cues for video classification. The approaches based on spatiotemporal networks have a huge amount of computing due to many trainable parameters tuned in the training phase. A component-based approach is proposed to represent the video content, weakly supervised learning (WSL) method [41], and requires less annotated data. A three-stream CNN is suggested that receives two representations and were fused with the motion-encoding stream. The LSTM block models each of the three streams' temporal relationship. For the fusion of the three streams and the final prediction generation, an fc layer is used.

The literature study showed that the best methods for activity understanding are the deep learning-based approach. An essential issue in these methods is much data for all recognizable classes required for model training. They are only able to detect activities seen by the model. Providing the training data for all possible HOI categories is costly and impractical. We focus on this problem and try to solve it through the zero-shot learning approach.

2.2. Zero-Shot Learning. Zero-shot learning is an exciting approach in different areas [42–45]. Most new methods based on zero-shot learning have two stages and focus on attributes [46–50]. The attributes are predicted in the first step and then infers class labels in the second step. The compositional learning for Visual Question Answering (VQA) has been explored [51], in which the VQA task breaks down into a sequence of modular subproblems. Each subproblem is modeled by one neural network.

For zero-shot action recognition, simultaneous object-action detectors training in the videos is suggested to identify object-action pairs [52], which uses the two-stream faster R-CNN [53], and one fc layer operates on both streams' concatenated features. This approach is not just for human action recognition and includes actions, such as "cat eatings" or "dog jumping." The attributes have also been used to understand human activities in an independent learning framework for recognizing objects and actions [54, 55]. The strong relationship between the objects and the actions is used for zero-shot recognition of action [56, 57].

HOI recognition through zero-shot learning is proposed in [13] that predicts the verb-object pairs from a still image. This method used a two-branch neural architecture that jointly trained for simultaneous recognition of objects and verbs. A similar approach is presented to zero-shot HOI recognition in video data [14]. An external knowledge graph is suggested [58] to validate predicted verb-object pairs and identify the most valid pairs. The external knowledge graph is made by extracting subject, verb, and object (SVO) triplets from knowledge bases [23, 59]. Each node in the graph is a verb or a noun (object), and its word embedding is the node's feature.

Our work is also scaling HOI recognition through zero-shot learning, but we focus on video data, which has more challenges. We present a neural architecture that can understand videos and detect objects and verbs in videos containing an HOI activity. We also proposed the use of side information to prevent predicting the invalid verb-object pairs (see Section 3.6).

2.3. Object Detection. Our proposed zero-shot learning method is compositional learning, in which the HOI decomposes into two components, verbs and objects. In other words, there are two components for recognition, verbs (human action) and objects. Recent advances in object detection have been achieved by the successful methods of region proposal [60] and region-based convolutional neural networks (R-CNN) [61]. Some works focused on processing time that is appropriate for real-time object recognition tasks. Only one processing step for recognizing the object in the image is suggested (YOLO) that concentrates on processing time [62]. Single-shot detector (SSD) [63] presented high-speed multiobject detection that uses different feature maps extracted from different layers of CNN to detect objects and their location in the image with varying sizes.

3. Proposed Approach

The primary purpose of this work is the ability to identify a novel HOI. We use the zero-shot learning approach because it increases identifiable HOIs without increasing HOI categories in training data. In other words, to train a recognizer model for a given number of classes, part of the target classes' data is sufficient. It is not necessary to have the data of all categories. Therefore, the need for training data is reduced. In this work, the input is a video (sequence of frames) containing a human-object interaction, and the output is a pair of "verb, object" as an HOI label.

Reducing the number of invalid predicted HOI classes, which are probably incorrect, is another goal of this work. For this purpose, the use of external information is suggested.

3.1. Zero-Shot on HOI Recognition. In zero-shot learning, the main class is decomposed into its components, and components recognition is applied instead of recognizing the main category. Identifying a new class in the zero-shot learning approaches is done by recognizing the class' components, which have been present separately in other classes seen by the model. In the test phase, the class components are recognized, and the predicted class is identified as a combination of the predicted components. Thus, a new combination of components, which the model did not see at the time of training, indicates the identification of a new class and is not labeling as a more similar existing class. Decomposing an HOI into a verb and object is previously introduced to identify the limited number of HOIs in still images [13]. Each HOI class decomposes into a verb and an object as its components. A particular verb can be performed on several different objects, for example, "writing on the whiteboard," and "writing on the notebook." Different verbs can also be performed on the same objects, such as "writing on the notebook" and " reading a notebook." If the system learns verb and object classes separately instead of HOI classes, it can recognize those verbs and objects in seen and unseen HOI classes and make verb-object pairs as an HOI class at test time. Suppose a particular object learned by a model from an HOI (with a specific verb) and that object exist in another HOI (with a different verb). In that case, the model can identify it, and it is not necessary to learn this object to model by second HOI class. The same applies to verbs. In other words, it is not required to feed all HOI categories to model for understanding all of them, and it only needs to have a training dataset, which includes all verbs and all objects. In other words, the problem of HOI recognition is decomposed into two recognition issues: verb recognition and object recognition. Therefore, the designed system must include two separate parts: verb recognition and object recognition.

Suppose the available HOI dataset includes v verbs and o objects. So, the identification system can recognize v verbs and o objects. Since an HOI class consists of a verb and an object, this system theoretically can identify $|v|.|o|$ categories. Training a recognizer system for understanding $|v|.|o|$ classes in a fully supervised manner needs to labeled training data for $|v|.|o|$ HOI classes. But in the proposed approach, we only need labeled data for $|v| + |o|$ categories. Also, since an HOI has one verb and one object, it can train both verb and object.

According to the above, the central system has two branches due to recognizing two components, namely, verb and object. The output of one branch is the predicted verb applied to the object, and the production of another branch is the object(s), which a verb is applied to it. The combination of the two outputs can be considered as a predicted HOI class.

The zero-shot learning methods have two stages: (1) predicting the components and (2) inferring the class label from predicted parts. The first stage in our problem is predicting the verb and objects, and the second is combining object and verb to infer HOI class. For the first stage, we use a two-branch structure that predicts the verb and objects. The second stage is done using side information to form an HOI class with verbs and objects obtained in the first stage.

The main idea of zero-shot on HOI recognition is introduced for the limited number of classes in still images [13]. Understanding video and, in particular, understanding HOI in video data is more challenging and more applicable than still image data, since, in this work, the zero-shot on HOI recognition in the video data is desired. Since the HOI is decomposed into two components (verb and object), the central recognizer system includes two main branches as two recognition tasks: one branch for verb recognition and one branch for object recognition. In this work, each recognition task is implemented by a neural structure. Figure 1 shows the simple architecture for the mentioned system. The verb recognition branch uses the RGB frames and optical flow of input video, while the object recognition branch uses only RBG frames for detection. In each branch, the input video (RGB frames and/or optical flow) feeds to the CNN module for extracting high-level features, and then these features are used for the corresponding recognition tasks. The object recognition branch is more straightforward because it can be recognized from a single frame. So, we use a typical CNN-based object recognition structure. But the nature of verb recognition is more complicated than the object. To recognize the verb, we use a three-stream structure based on CNNs and RNNs.

3.2. *Object Recognition Branch.*

Our focus and innovation are on the verb recognition branch. The object recognizer's desired output is the recognized objects from the input video and their reliability score. The objects of each frame of the input video are recognized by the existing successful object recognition method, SSD [63]. The SSD approach is based on the feed-forward CNN that produces a fixed-size collection of bounding boxes. After that, the score of object class instances in those boxes is predicted, and a nonmaximum suppression step makes the final detections. SSD is a fast object detection method because of eliminating bounding box proposals and subsequent feature resampling stage. The early network layers are based on standard architecture used for high-quality image classification. Some convolutional feature layers are added to the previous layers, which decrease in size progressively and allow predictions of detections at multiple scales. SSD uses separate predictors (filters) for different aspect-ratio detections. These filters apply to multiple feature maps to perform detection at multiple scales. So, the location of objects in an image and their reliability scores are predicted in a short time. See reference [63] for more details. In this work, we have not used the location information of the objects, and we have considered only the detected objects along with their score. Still, in future works, we can use the location information of the objects and salience areas of action to distinguish the target object from the background objects.

After detecting objects in each frame by SSD, the objects are obtained in the whole video. These objects combine with the recognized verb using side information (see Section 3.6) and reliability scores to make a valid verb-object pair, and the HOI class is identified.

3.3. *Verb Recognition Branch.*

Verb recognition or, in general, activity understanding in video space is different from single image space. For activity-based video classification in deep learning approaches, usually, the features of each frame of input video extracted with a neural structure and class of video clip predicted by a set of features came from all frames. The use of two-stream structures is common

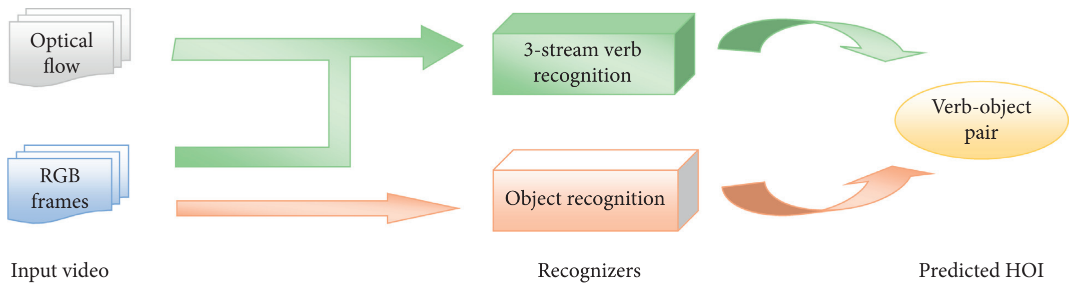


Figure 1: A simple overview of the main system architecture. The verb recognition branch uses the RGB frames and optical flow of input video, while the object recognition branch uses only RBG frames for detection.

for this purpose, in which one stream considers appearance cues (from RGB frames) and the other considers temporal cues (from optical flow). One input to activity understanding systems usually is a sequence of RGB frames of an input video clip. Much of an RGB image is the background and is not necessarily related to the activity. Hence, the features extracted from it are strongly affected by the background. Estimating susceptible areas to activity and extracting features from them can help us solve this problem. On the other hand, the background can also contain information about the event that occurred, and completely removing it can lead to performance degradation. Estimating the region related to activity and blurring the background can be useful. Thus, the RGB stream is split into two streams. The first stream estimates activity region patches and extracts features from them as patch-based representation. The second stream estimates the activity region, blurts other areas, and extracts features from the new RGB image as focal representation. The video data includes temporal information of what is happening, which is not in still images. The short-time temporal information can be represented by optical flow. Therefore, a common input to activity understanding systems can be an optical flow of input video for motion representation. These processing streams are described more detail in Section 3.5.

Given the above and that the nature of the verb's recognition is a subset of the action, we propose to use three inputs for the verb recognition task. These three inputs are estimated activity region patches, RGB image with blurred background, and the optical flow. So, the central system's verb recognition branch is a three-stream structure, including patch-based representation, focal representation, and motion representation. This three-stream structure was previously introduced for action recognition in the video [41]. We use this structure with a new feature aggregation technique to recognize the verb in the video.

3.4. Feature Aggregation.

3.4. Feature Aggregation. The final step in the recognition system is classification on a feature vector derived from three processing streams. Features obtained from each stream must aggregate to produce the final feature vector of each stream, and then the ultimate features of the overall system for classification are obtained. Conventional feature aggregation methods, such as average or max-pooling, represent the entire space of features as a single descriptor. These methods may be suboptimal to representing a video containing several subactions. Locally aggregation features were introduced in [64] and extended to spatiotemporal feature aggregation for action recognition as Action VLAD [7]. In this scenario, the features are clustered to K cluster and pooled jointly across space and time. Figure 2 shows the difference between spatiotemporal and average or max-pooling aggregation. In the average and max pooling scenarios (Figures 2(a) and 2(b)), the entire space of the feature map is represented as a single descriptor. But in the Action VLAD scenario (Figure 2(c)), the feature space is represented by several (K) descriptors. With this technique, if the nature of the action consists of several sub-actions, we hope that it will be described more optimally. Therefore, it is more likely to recognize the correct action because the feature space is represented by multiple descriptors instead of one, and the deletion of information is less in the feature aggregation step.

Consider the extracted descriptors from each frame of the video in each spatial location be $x_{i,t} \epsilon R^D$, where $i \epsilon \{1 \ldots N\}$ is related to spatial location and $t \epsilon \{1 \ldots T\}$ is the frame index. For spatiotemporal aggregation, the descriptor space R^D is divided into K cells using K anchor points $\{c_k\}$ (stars in Figure 2(c)). Then, each descriptor $x_{i,t}$ was assigned to one of the cells due to its distance from the anchor. The new descriptor is presented by the difference vectors calculated across the entire video as follows.

$$V[j,k] = \sum_{t=1}^{T} \sum_{i=1}^{N} \frac{e^{-\alpha \left\| x_{i,t} - c_k \right\|^2}}{\sum_{k'} e^{-\alpha \left\| x_{i,t} - c_{k'} \right\|^2}} \left(x_{it}[j] - c_k[j] \right), \quad (1)$$

where $x_{it}[j]$ and $c_k[j]$ are the j-th component of the descriptor vector $x_{i,t}$ and anchor c_k. Parameter α is a tunable hyperparameter. The output is a matrix V where each column shows an aggregated descriptor related to one cell. The matrix intranormalized across columns, stacked, and L2-normalized [65] into a single descriptor of the entire video.

The difference vectors record the differences of extracted descriptors from subactions represented by anchors c_k. So, this aggregating scenario can help to recognize the verbs that consist of some subactions.

3.5. Details of Each Stream in Verb Recognition. Our proposed model for verb recognition is a three-stream RNN-based structure. Each stream has three main processing

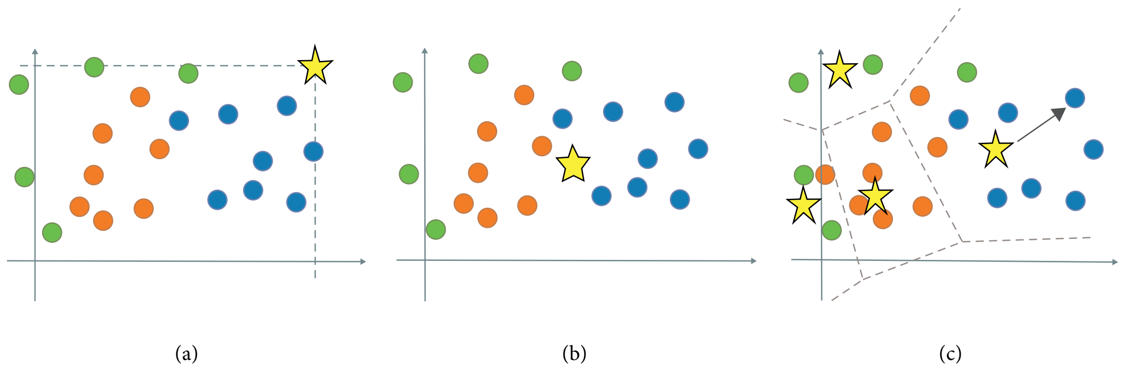


Figure 2: Difference pooling scenario for aggregate features. Different colors points correspond to different subactions in the video. (a) and (b) are good for similar features, but they do not adequately capture the complete distribution of features if the input video contains several subactions. Scenario (c) clusters features in spatiotemporal manner [59].

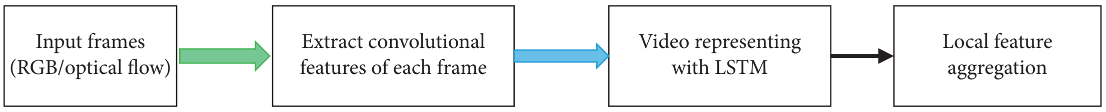

Input frames (RGB/optical flow)
Extract convolutional features of each frame
Video representing with LSTM
Local feature aggregation

Figure 3: Block diagram of the process in each processing stream in the verb recognition branch shown in Figure 1. First, the convolutional features of each frame were extracted. Then, the whole input video is represented by the LSTM block. Finally, the elements are locally aggregated.

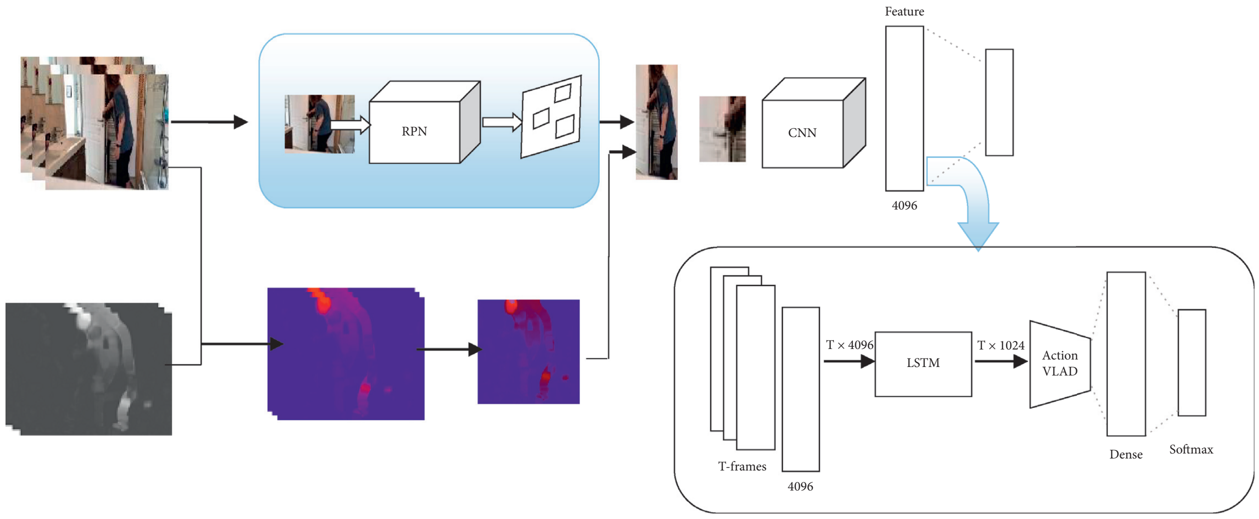


Figure 4: Patch-based representation. At first, the areas related to the target verb are detected, and the patches are extracted from the input frame. Then, the features of each patch in each frame are extracted. The LSTM block represents the whole input video. Finally, the elements are locally aggregated, and the class scores for each verb class are estimated.

steps, which are shown in Figure 3. The processing flow of the three streams is almost similar. Each stream's input is a sequence of frames in the form of RGB and/or optical flow. CNN extracts the convolutional features of each frame. Then, the extracted features of the T consecutive frames of video feed to LSTM blocks to represent the temporal information. The output is several spatiotemporal feature vectors. These vectors are then locally aggregated, and the final representation vector of each stream is prepared for the final classification. The following is a detailed description of each shown module in Figure 3.

3.5.1. Path-Based Representation. This stream aims to find areas related to the target verb and use it to identify the verb. These regions are appropriate to represent the video clip based on the event that occurred in it. Figure 4 shows the structure and processing process in this stream. The probable areas are selected using the method proposed by Papazoglou et al. [66], which uses the RGB frame and its optical flow. Other proposed regions are taken from the region proposal network (RPN) offered by Ren et al. [53]. The RPN extracts the areas that are prone to the presence of objects or entities. The RPN processes a still image and outputs lots of proposal windows, many of which are irrelevant. The final action patches were selected by merging the already and previously extracted regions. We see Algorithm 1 in [41] to choosing actionness patches process (see Algorithm 1 for more details). After selecting each frame's actionness patches, these are fed to CNN, and the convolutional features are extracted. All of these processes are related to the second block in Figure 3. After obtaining convolutional features of all T frames of the input clip, T feature vectors are fed to the RNN block and outputs T time-distributed feature vectors as the input video's temporal representation (third block in Figure 3). Of course, we use the LSTM block as an RNN for the video's temporal representation in all three streams. The final step in this stream is local feature aggregation (see Section 3.4). The time-distributed features are locally aggregated to the K descriptor, and the target activity is represented as its subactions. The output vectors are used to classify the occurred verb in the video clip, and the classification scores are predicted for each verb class.

3.5.2. Focal Representation. As previously stated, much of an RGB image is the background and is not necessarily related to the event. So, it may lead to overfitting in the training phase if the background is not discarded. On the other hand, completely removing the background can lead to performance degradation because it can also contain information about the event. To handle this issue, after finding the foreground (selecting the probable area in patch-based representation stream), the background of the RGB image is blurred by a Gaussian low pass filter, and the other areas remain unchanged. The idea is inspired by the human focal vision system [41]. The resulting image is a focused image on the area prone to activity, which also retains background information. Subsequent steps, including convolutional feature extraction from all T frames, obtain time-distributed feature vectors using LSTM block, local feature aggregation, and predicting the classification scores, are quite similar to patch-based representation stream' steps. Figure 5 shows the structure of this processing stream.

3.5.3. Motion Representation. According to the contents of Section 3.3, the short-term temporal information in the video clip can be represented by optical flow. The RNN block can obtain long-term temporal information. So, the third stream of the verb recognition branch can be motion representation by optical flow (Figure 6). Each frame's convolutional features are extracted from optical flow by a CNN. The structure used for this stream is the motion-CNN proposed in [31]. The optical flow is computed between each consecutive frame using the Brox algorithm [67], which assumes the camera is static. As shown in Figure 6, the next steps are exactly like the two other streams.

As observed, the processing flow is the same in all three streams. Only the inputs of these streams are different. The first stream uses the salience patches of the input image, the second stream uses a focal image whose background is blurred, and the third stream uses optical flow. The output of each stream is classification scores for verb classes. Finally, these three streams' results are merged to recognize the target verb in the input video.

3.6. Side Information for Reducing Invalid HOIs. The zero-shot learning approach has two stages: (1) predicting the components and (2) inferring the class label from predicted parts. The first stage of this approach in our work is to recognize the verbs and objects done by the central system (two-branch HOI recognition system). The second stage is not complicated. It is enough to put the recognized verb and the object together and create the "verb-object" pair as a predicted HOI. But is any combination of verb-object acceptable? For example, the "eating a laptop" is a presumable verb-object combination that may be the central system's output. Is it acceptable? Of course not. So, there is a need for a scenario to solve this problem. We also tackle this problem in this work.

Many of our interactions with objects are based on our prior knowledge. We know that a s "laptop" is not edible, and we cannot eat it. Hence, we argue that the detected pair of verb-object (eating a laptop) is invalid. This argument is based on our prior knowledge. If the system has prior knowledge like humans, it can validate the output pairs and identify invalid states. In this case, the system realizes that "eating a laptop" is an incorrect HOI and seeks another verb or object to create a valid HOI.

The use of an external information graph is proposed for compositional learning for HOI [58]. The idea of using the side information comes from the concept of word embedding. The external graph encodes two essential types of knowledge: (1) the "affordance" of objects, such as "laptop can be held," and (2) the semantic similarity between verbs or objects. SVO triplets define objects' affordance from the external knowledge base [59], and the similarity between verbs or objects is defined by lexical information from WordNet [68].

We propose a simple graph to modeling and using side information (Figure 7). The graph has three categories of nodes: verb, object, and interaction. Each verb and object is modeled as a separate node, and their attributes are provided from nltk [69] based on the concept of word embedding [70, 71]. These attributes are conceptual representations of words so that words with close meanings have similar attributes. For example, both the words "Sandwich" and "pizza" are related to a type of food, so they have a similar concept and are used in a similar sense. A verb node can only connect to an object node via a valid interaction node and create a graph path. So, each path in this graph shows a valid HOI. Valid HOIs are HOIs that exist in the dataset. There is no path between verbs together or objects together. Also, conceptually similar verbs (or objects) are connected with a link. The similarity between verbs (or objects) is computed by nltk [69]. The links help in finding the valid secondary HOIs. For example, let "hold a laptop" is a valid HOI (it existed in the database and its path exists in the graph), and "take a laptop" has not a path in the graph (it did not exist in the dataset), but there is a

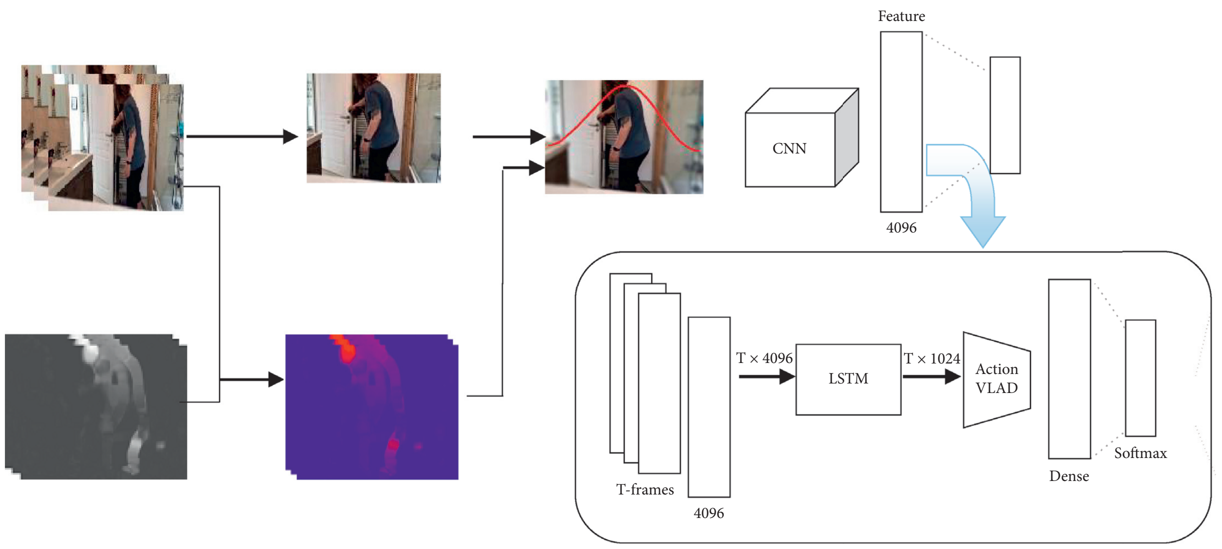


Figure 5: Focal representation. At first, the area related to the foreground is detected from the input frame, and the background is blurred with a lowpass Gaussian filter. Then, the features of each blurred frame are extracted. The whole input video is represented by the LSTM block. Finally, the elements are locally aggregated and the class scores for each verb classes are estimated.

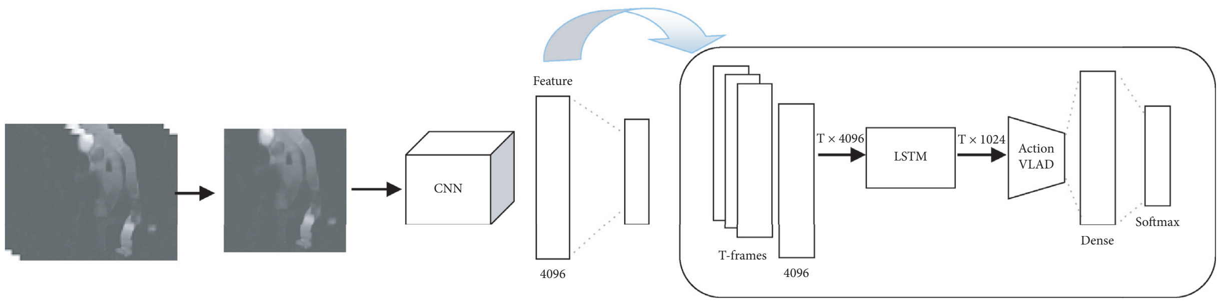


Figure 6: Motion representation. The short-term temporal information in the video clip can be represented by optical flow. In first, the features of each optical flow of each frame extracted. Then the whole input video is represented by the LSTM block. Finally, the elements are locally aggregated, and the class scores for each verb classes are estimated.

link within the “hold” and “take” nodes. Therefore, “take a laptop” can be a valid HOI (it is a valid secondary HOI). This rule also applies to object nodes.

The side information graph is used to validate the central system’s results and enhance the overall performance. The central system’s output is the classification score for the verb classes and the identified objects with their reliability score. The three verb classes with the highest classification score are combined with the identified objects to form possible “verb-objects ” pairs and are sorted by score. The validity of the obtained pairs is then checked using the side information graph, and the first valid pair is selected as the final predicted HOI class. In fact, this is the second stage of the zero-shot learning approach.

4. Results and Discussion

We present the results of our method in this section and compare it to some other works. The used dataset is introduced first, and then the implementation setups are described. Finally, we report our results and compare the proposed approach against state-of-the-art methods.

4.1. Dataset. For human action understanding in videos, several appropriate datasets have been provided and published, such as UCF101 [72], HMDB51 [73], and Actor-Action Dataset (A2D) [74]. Most of these datasets involve many human activities, not just HOIs. So, they are not suitable for the evaluation of HOI understanding tasks.

The recently published challengeable dataset for human activity understanding is Charades [12]. This dataset contains 9848 video clips of HOIs captured in real environments. It has 157 categories of human activities including some actions with “no interaction.s ” After excluding categories with “no interaction,s ” there are 149 valid HOI categories defined as verb-object pairs. This 149 category includes 34 verbs and 37 objects. Clips of this dataset cover both the third and first person’s actions. We use the third person’s clips of these 149 categories as our Charades benchmark.

We have two scenarios for evaluating our system. First, we assess the model for fully supervised HOI recognition and compare model performance with some state-of-the-art approaches. Afterward, we present the performance of the proposed model on the zero-shot detection of HOIs. For

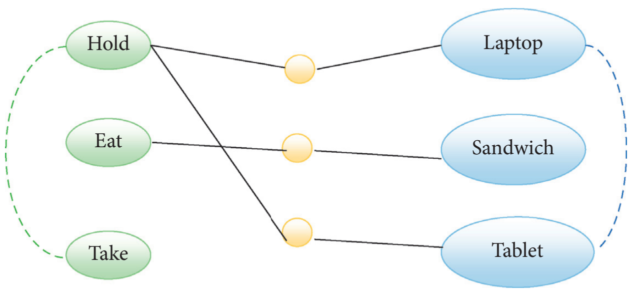


Figure 7: Side information graph. The green ellipse shows the verb nodes, the blue ellipse shows the object nodes, and the yellow ball shows an interaction. Each valid HOI is specified by the triple connected nodes (verb, interaction, and object). The conceptually similar verbs or objects nodes connected by a link (dashed lines).

zero-shot analysis, we split each set of verbs and objects into two subsets. The object set is divided into subsets 1 and 2, and the verb set is divided into subsets A and B. So, we can provide four subgroups of HOI, including 1A, 1B, 2A, and 2B. For example, subgroup 1A consists of 49 HOIs whose verbs are in the verb subset A, and their objects are in the object subset 1. The same applies to the other three subgroups. Subgroup 1B includes 22 HOI classes, 2A includes 47 HOIs, and 2B includes 31 HOIs.

If we train the model with 1A + 2B, it does not see all HOIs but see all verbs and objects. So, it can identify the unseen HOIs that are in the subgroups of 1A and 2B. In other words, we are using 80 HOI classes (1A + 2B) to train a system that can recognize 149 HOI classes of the used dataset.

4.2. Implementation Details. The proposed system has three processing streams for verb recognition and one stream for object recognition. For the two spatial CNN streams of the verb recognition branch, an AlexNet architecture that pretrained on UCF sports, JHMDB, and HMDB51 datasets, is used. The first spatial network inputs are the action patches, and for the second spatial network, the proposed focal representation is fed. Moreover, a VGG16-RPN, which is trained on the ImageNet dataset, is used for region proposal to select the actionness patches process. For the 3rd stream of the verb branch as motion representation, we used the CNN network like the network architecture used by Gkioxari et al. [31]. This motion-CNN is pretrained on the optical flow images of UCF sports and JHMDB datasets. The optical flow is computed between each consecutive frame using the Brox algorithm [67]. For motion-CNN input, a 3D image is created by stacking the x-component, y-component, and optical flow magnitude. The FC7 layer of three CNNs extracts a 4096-dimensional feature vector for each input video frame. After obtaining feature vectors for all T frames of input clip in three CNNs, these feature vectors are fed to the RNN block and outputs T time-distributed feature vectors. For the RNN block, the LSTM module with 1024 hidden units is used. The last step before the final classification is local feature aggregation (see Section 3.4), in which the value 64 is selected for parameter K. The time-distributed features are locally aggregated with ActionVLAD, and the target activity is represented as its subactions. The output of this step is used to classify the occurred verb in the video clip. Two FC layers with the number of neurons equal to 256 and the number of verbs (here 34) are used as a classifier in each stream. For training the LSTM and its following dense network, a stochastic gradient descent optimizer (SGD) is utilized. The last FC layer determines the final prediction with a Softmax activation. For preventing overfitting, the flipping video frames technique is used for data augmentation. The learning rate is set to value 5×10^{-5}. Also, we use $T = 25$ frames per video for both optical flow and RGB for learning and evaluation. The final verb class scores are obtained by averaging the three streams' results.

Another processing branch of the main system is the object recognizer. In this branch, the objects of each frame of input video are recognized by the existing successful object recognition method, SSD [63]. So, the objects are obtained in the whole input video. The results are objects with their reliability scores. These results combine with the recognized verb by using side information (see Section 3.5), and a valid verb-object pair is identified as the HOI class. Our deep learning system is implemented in python based on the Tensorflow open-source toolbox and Keras library.

4.3. Experimental Results. We start the experiments by comparing the zero-shot recognition accuracy of our initial model and the state of the art. The initial model has two spatial processing streams in the verb recognition branch (without motion representation). Table 1 shows the results. The effect of using side information has also been investigated. The two last rows in Table 1 show the results of our simple system with or without side information (SI). The compared methods are all in the field of zero-shot learning, and, like us, they have tried to identify unseen classes. Our previous model [14] has one stream in verb branch recognition. The method [58] uses the convolutional graph networks, which learn how to compose classifiers for verb-noun pairs. The SES [68] and DEM [75] use the verb and noun embeddings, which are matched to visual features using L2 loss. CC [76] does not combine word embeddings but considers the composition of classifiers.

Our model better represents the video due to RNN blocks' use, which leads to better verb recognition. So, it has had better results. The use of side information graphs also had a positive effect on the results in Table 1. In Figure 8, two samples showed that they were misclassified without using the side information and were classified correctly after using the side information. The patterns of the right verbs and the false detected verbs are similar. Therefore, the model may classify incorrectly, but using the side information can correct such errors.

We propose using local feature aggregation to aggregate the feature maps extracted from input frames (Section 3.4). The proposed model for this evaluation is named 2Stream + WE + VLAD. The effect of this technique is shown in Table 2. The reported results indicate a slight performance improvement. We used the local aggregation after the RNN module, but it is possible to apply this technique to the

Table 1: Zero-shot HOI recognition mAP on Charades dataset. The model trained with 1A + 2B and tested on 2A + 1B and all data.

Method	mAP (%) on the test set	
	All data	Unseen data (2A + 1B)
Chance	1.43	1.45
Compositional [58]	14.32	10.48
SES [68]	13.12	9.56
DEM [75]	11.78	8.97
CC [76]	14.31	10.13
1stream [14]	16.48	11.23
2Stream – SI	**17.8**	**14.83**
2Stream + SI	**19.5**	**16.08**

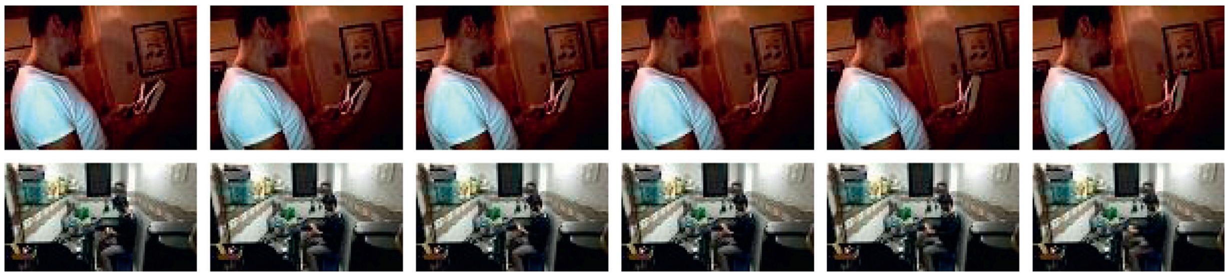


Figure 8: Two samples were misclassified without using side information and were classified correctly after using the side information. In the first row, the true HOI class is "smiling at a book." Without using the side information, the predicted class was "playing, book." After using the side information, the predicted class is "smiling, book." In the second row, the true HOI class is "making a sandwich.s " Without using the side information, the predicted class was "fixing, sandwich." After using the side information, the predicted class is corrected as "making, sandwich."

Table 2: The effect of local feature aggregation on HOI recognition performance. The model was trained with 1A + 2B and tested on 2A + 1B and all data.

Method	mAP (%)	
	ALL data	Unseen data (2A + 1B)
2Stream – WE - VLAD	17.8	14.83
2Stream + WE - VLAD	19.5	16.08
2Stream – WE + VLAD (rnn)	18.7	15.33
2Stream + WE + VLAD (rnn)	20.96	16.96
2Stream + WE + VLAD (cnn)	20.65	16.65

Table 3: The impact of the optical flow on the proposed system's performance. The model trained with 1A + 2B and tested on 2A + 1B and all data.

Method	mAP (%)	
	ALL data	Unseen data (2A + 1B)
2Stream – WE-VLAD	17.8	14.83
3Stream – WE-VLAD	**19.21**	**16.65**
2Stream + WE-VLAD	19.5	16.08
3Stream + WE-VLAD	**20.84**	**17.32**
2Stream + WE + VLAD (rnn)	20.86	16.96
3Stream + WE + VLAD (rnn)	**21.27**	**17.63**

Table 4: The impact of the RNNs (LSTM/GRU) on the proposed system's performance. Training and testing are performed on the same subset (averaged on four subsets).

Method	mAP (%)	
	LSTM	GRU
2Stream + WE - VLAD	20.28	20.33
3Stream + WE - VLAD	20.94	20.94
2Stream + WE + VLAD (rnn)	20.74	20.78
3Stream + WE + VLAD (rnn)	21.31	21.35

Table 5: The impact of the RNNs (LSTM/GRU) on the proposed system's performance. The model is trained with 1A + 2B and tested on all data.

Method	mAP (%)	
	LSTM	GRU
2Stream + WE - VLAD	19.5	19.42
3Stream + WE - VLAD	20.84	20.63
2Stream + WE + VLAD (rnn)	20.86	20.64
3Stream + WE + VLAD (rnn)	21.27	21.19

outputs of the CNNs. The results show that its application to the RNN module has a slight performance improvement.

For the representation of the temporal information in the input video, the potential of recurrent neural networks (RNNs) has been exploited. Furthermore, the use of the optical flow of the input video as the 3rd stream in the verb recognition branch has been investigated. The impact of using this stream is shown in Table 3. The model named **3Stream + WE + VLAD(rnn)** is our final proposed system, which uses the LSTM block as the RNN module. According to the results, using optical flow is observed in the slight improvement of system performance.

We used the RNNs for the representation of the temporal information in the input video. In previous evaluations, the LSTM block is used for the RNN module. Another choice is the GRU blocks, which are simpler than LSTMs

Table 6: The impact of the RNNs (LSTM/GRU) on the proposed system's performance. Training and testing are performed on all data.

Method	mAP (%)	
	LSTM	GRU
2Stream + WE - VLAD	22.45	22.20
3Stream + WE - VLAD	23.76	23.52
2Stream + WE + VLAD(rnn)	23.64	23.43
3Stream + WE + VLAD(rnn)	24.73	24.58

Table 7: HOIs recognition results (mAP(%)) on Charades dataset. The model sees all HOI classes in the training phase.

Method	mAP (%)	
	RGB	RGB + optical flow
ActionVLAD [7]	17.6	21.0
Sigurdsson et al. [8]	18.3	22.4
CoViAR [9]	21.9	24.1
Ours	**23.64**	**24.73**

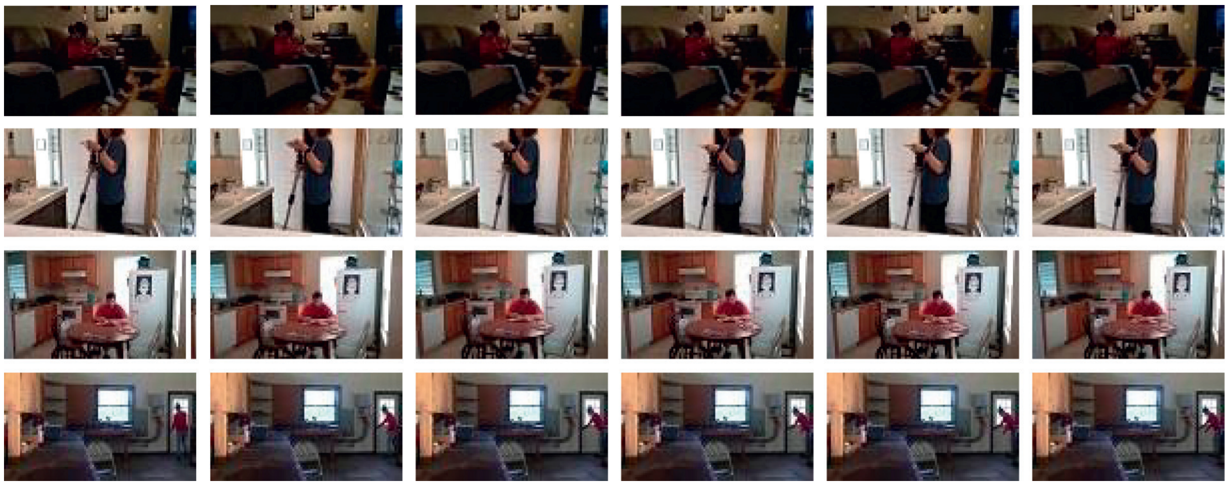


Figure 9: Such samples of incorrect classification of our final model. In the first row, the true class is "opening a laptop" but predicted as "fixing a laptop." In the second row, the class of "fixing a vacuum" was predicted as "holding a vacuum." Row 3 shows the "working at a table" that is predicted as "watching at a book," and the final row shows the "grasping onto a doorknob," which is predicted by our model as "fixing a door."

and have a similar function. The GRU blocks are used with 1024 hidden units. The comparison between the use of the LSTMs and GRUs is made, taking into account the volume of training data. The results are shown in Tables 4–6. The values reported in Table 4 indicate that the GRU has a slight improvement to the LSTM. But looking at the results of Tables 5 and 6 shows the opposite. The difference is the amount of training data. In other words, the GRUs are simpler than the LSTMs, and they converge faster. So, we conclude that the GRU is appropriate for cases that the amount of training data is small (due to the rapid convergence), and the LSTM is the right choice for cases with a large amount of training data due to its excellent performance.

For the last evaluation, we examine our proposed system in a fully supervised scenario. In other words, the model sees all HOI classes in the training phase. 80% of all videos were used as training data, and the remaining 20% of videos were used for testing. Also, 10% of the training data are used as the validation set. We compared our model's performance to three other state-of-the-art action recognition methods on the Charades dataset. These three approaches are ActionVLAD [7], Sigurdsson et al. [8], and CoViAR [9], which are DNN-based. Our method in this evaluation for RGB input is 2Stream + WE + VLAD and for RGB + optical flow is 3Stream + WE + VLAD. According to the result of the previous evaluation, the LSTM has been selected for the RNN module. The results are reported in Table 7, which shows our method's better performance compared to the other three methods.

5. Discussion

This work's primary goal is to identify HOI classes that the model had not seen before. The main idea is to decompose the HOI into verb-object pairs and recognize them independently. Tables 1–3 compare our proposed system to the state of the art from the perspective of the intended purpose. Using the side information, representing the video by 3-stream structure and the RNN blocks, and using the local feature aggregation have ultimately led to our system's better performance. Using the side information has corrected some invalid misclassifications (see Figure 8.). Using the local feature aggregation technique leads to a better representation of some of the classes that consist of several subactions.

Figure 9 shows some misclassified samples of our final system. Observing these misclassified examples shows the visual patterns of the predicted classes are similar to the actual classes. These errors indicate that a lack of educational data for different categories has made the model unable to learn a general pattern. For example, the "opening" class has several patterns: opening the door, opening the refrigerator, opening the cabinet, and opening the laptop. If the model learns the pattern of "opening" for the class of opening the door, it can predict the "opening" in opening the cabinet without observing it during training. But the visual pattern of the "opening" of opening a laptop is different, and the model cannot predict correctly. If the training data for a verb exists in other cases, the model can learn a more general pattern and perform better during testing.

In addition to the zero-shot performance, which was the primary purpose of the work, we evaluate our method in a fully supervised scenario and compare it to some methods (Table 7). In this case, the model sees all HOI classes in the training phase. Our system's performance is slightly better due to its potential in video representation and the correction of some errors.

6. Conclusion

In this research, we propose a CNN-based system for HOI understanding in video data through a zero-shot learning approach, which can identify new classes that have not been seen before. So, the proposed method can identify more HOI classes than available HOIs for training and partly resolve data unavailability for all possible HOI classes. Our approach decomposes the HOIs to verbs and objects and addresses the problem as verb and object recognition in the videos. The model has a two-branch neural structure for two recognition tasks, and it uses a CNN for feature extraction. We showed that we could use 80 HOI classes (1A + 2B) to train a system that can recognize 149 HOI classes of the used dataset (1A + 2B + 2A + 1B). Of course, there can be more predictable classes in the real world because not all possible real-world combinations of objects and objects are in this used dataset. In other words, the information and potential of the available data can be better used.

We proposed using a local feature aggregation to better represent verbs (actions), especially verbs with multi-subaction, before final classification. Conventional feature aggregation methods represent the entire space of features as a single descriptor, which may be suboptimal to representing a video containing several subactions. The used local feature aggregation technique prevents the deletion of information when merging features. So, the recognition of verbs with several subaction is improved.

We also proposed using side information to reduce the prediction of invalid verb-object combinations. Because of the separate recognition of verb and object, predicting the invalid verb-object pairs is possible. The side information shows the relations between verbs and objects defined by lexical information.

We showed the effect of each proposed technique on HOI recognition system performance. We also showed that our method could work slightly better than some fully supervised HOI recognition methods that reported the best results on the used dataset, although this improvement is tiny.

A more appropriate structure can be provided with better accuracy in recognizing the verb from the input video for future work. The distinction between background objects and activity-related objects can be used in future work, taking into account the location of the detected objects. We will also work on updating the system to learn new verbs or object classes without the need for data from previous classes. In other words, it is possible to apply incremental learning to the proposed system.

Data Availability

Previously reported image data were used to support this study and are available at https://doi.org/10.1007/978-3-319-46448-0_31. These prior studies (and datasets) are cited at relevant places within the text.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] B. Yao and L. Fei-Fei, "Modeling mutual context of the object and human pose in human-object interaction activities," in *Proceedings of the Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 17–24, San Francisco, CA, USA, April 2010.

[2] J.-F. Hu, W.-S. Zheng, J. Lai, S. Gong, and T. Xiang, "Recognizing human-object interaction via exemplar-based modeling," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3144–3151, Sydney, Australia, December 2013.

[3] Y.-W. Chao, Z. Wang, Y. He, J. Wang, and J. D. Hico, "A benchmark for recognizing human-object interactions in images," in *Proceedings of the IEEE International Conference on Computer Vision*, Long Beach, CA, USA, June 2015.

[4] A. Mallya and S. Lazebnik, "Learning models for actions and person-object interactions with transfer to question answering," in *Proceedings of the Computer Vision – ECCV 2016: 14th European Conference*, pp. 414–428, Amsterdam, The Netherlands, October 2016.

[5] Y.-W. Chao, Y. Liu, X. Liu, H. Zeng, and D. Jia, "Learning to detect human-object interactions," in *Proceedings of the 2018 IEEE Winter Conference on Applications of Computer Vision (Wacv)*, pp. 381–389, Lake Tahoe, NV, USA, March 2018.

[6] G. Gkioxari, R. B. Girshick, P. Doll' ar, and K. He, "Detecting and recognizing human-object interactions," 2017.

[7] R. Girdhar, D. Ramanan, A. Gupta, J. Sivic, and B. Russell, "Actionvlad: learning spatio-temporal aggregation for action classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 971–980, Honolulu, HI, USA, July 2017.

[8] G. A. Sigurdsson, S. Divvala, A. Farhadi, and A. Gupta, "Asynchronous temporal fields for action recognition," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, Honolulu, HI, USA, July 2017.

[9] C.-Y. Wu, M. Zaheer, H. Hu, R. Manmatha, A. J. Smola, and P. Krähenbühl, "Compressed video action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6026–6035, Salt Lake City, UT, USA, June 2018.

[10] C. Gao, Y. Zou, and J. Huang, "ican: Instance-centric attention network for human-object interaction detection," *arXiv:1808.10437*, 2018.

[11] Y. Li, S. Zhou, X. Huang et al., "Transferable interactiveness knowledge for human-object interaction detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3585–3594, Long Beach, CA, USA, 2019.

[12] G. A. Sigurdsson, G. Varol, X. Wang, A. Farhadi, I. Laptev, and A. Gupta, "Hollywood in homes: crowdsourcing data collection for activity understanding," in *ECCV 2021: European Conference on Computer Vision*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., pp. 842–856, Springer, Berlin, Germany, 2016.

[13] L. Shen, S. Yeung, J. Hoffman, G. Mori, and L. Fei-Fei, "Scaling human-object interaction recognition through zero-shot learning," in *Proceedings of the 2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1568–1576, Lake Tahoe, NV, USA, March 2018.

[14] V. O. Maraghi and K. Faez, "Zero-shot learning on human-object interaction recognition in video," in *Proceedings of the 2019 5th Iranian Conference on Signal Processing and Intelligent Systems (ICSPIS)*, pp. 1–7, Shahrood, Iran, July 2019.

[15] M. Ziaeefard and R. Bergevin, "Semantic human activity recognition: A literature review," *Pattern Recognition*, vol. 48, no. 8, pp. 2329–2345, 2015.

[16] J. J. Gibson, "The ecological approach to visual perception," 1979.

[17] L. Stark and K. Bowyer, "Achieving generalized object recognition through reasoning about association of function to structure," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 10, pp. 1097–1104, 1991.

[18] V. Delaitre, D. Fouhey, I. Laptev, J. Sivic, A. Efros, and A. Gupta, "Scene semantics from long-term observation of people," *International Journal of Computer Vision*, vol. 110, no. 3, pp. 259–274, 2014.

[19] A. Gupta, A. Kembhavi, and L. S. Davis, "Observing human-object interactions: using spatial and functional compatibility for recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 10, pp. 1775–1789, 2009.

[20] V. Delaitre, J. Sivic, and I. Laptev, "Learning person object interactions for action recognition in still images," *Advances in neural information processing systems*, vol. 54, pp. 1503–1511, 2011.

[21] S. Maji, L. Bourdev, and J. Malik, "Action recognition from a distributed representation of pose and appearance," in *Proceedings of the Computer Vision and Pattern Recognition (CVPR)*, pp. 3177–3184, Colorado Springs, CO, USA, April 2011.

[22] C. Desai and D. Ramanan, "Detecting actions, poses, and objects with relational phraselets," in *Proceedings of the European Conference on Computer Vision*, pp. 158–172, Florence, Italy, May 2012.

[23] R. Krishna, Y. Zhu, O. Groth et al., "Visual genome: connecting language and vision using crowdsourced dense image annotations," *International Journal of Computer Vision*, vol. 123, no. 1, pp. 32–73, 2017.

[24] B. Fernando and S. Gould, "Learning end-to-end video classification with rank-pooling," in *International Conference on Machine Learning*, pp. 1187–1196, New York, NY, USA, June 2016.

[25] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 568–576, Montreal, Canada, June 2014.

[26] S. Ji, W. Xu, M. Yang, and K. Yu, "3D convolutional neural networks for human action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221–231, 2012.

[27] G. Chéron, I. Laptev, and C. Schmid, "P-cnn: pose-based cnn features for action recognition," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3218–3226, Santiago, Chile, December 2015.

[28] J. Carreira, A. Zisserman, and Q. Vadis, "Action recognition? A new model and the kinetics dataset," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4724–4733, Salt Lake City, UT, USA, June 2018.

[29] C. Lu, R. Krishna, M. Bernstein, and L. Fei-Fei, "Visual relationship detection with language priors," in *Proceedings of the European Conference on Computer Vision*, pp. 852–869, Amsterdam, Netherlands, October 2016.

[30] A. Karpathy, T. George, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1725–1732, Columbus, OH, USA, June 2014.

[31] G. Gkioxari and J. Malik, "Finding action tubes," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 759–768, Boston, MA, USA, March 2015.

[32] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1933–1941, Las Vegas, Nevada, USA, July 2016.

[33] C. Feichtenhofer, A. Pinz, R. P. Wildes, and A. Zisserman, "What have we learned from deep representations for action recognition?" in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7844–7853, Salt Lake City, UT, USA, March 2018.

[34] L. Wang, Y. Qiao, X. Tang, and L. Van Gool, "Actionness estimation using hybrid fully convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2708–2717, Las Vegas, NV, USA, June 2016.

[35] X. Peng and C. Schmid, "Multi-region two-stream R-CNN for action detection," in *Proceedings of the European Conference on Computer Vision*, pp. 744–759, Amsterdam, The Netherland, October 2016.

[36] M. Zhang, C. Gao, Q. Li, L. Wang, and J. Zhang, "Action detection based on tracklets with the two-stream CNN,"

Multimedia Tools and Applications, vol. 77, no. 3, pp. 3303–3316, 2018.

[37] P. Weinzaepfel, Z. Harchaoui, and C. Schmid, "Learning to track for spatio-temporal action localization," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3164–3172, City Park, Chile, June 2015.

[38] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4489–4497, Santiago, Chile, August 2015.

[39] G. W. Taylor, R. Fergus, Y. LeCun, and C. Bregler, "Convolutional learning of spatio-temporal features," in *Proceedings of the European Conference on Computer Vision*, pp. 140–153, Berlin, Germany, July 2010.

[40] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised learning of video representations using lstms," in *Proceedings of the International Conference on Machine Learning*, pp. 843–852, Lille, France, July 2015.

[41] V. Adeli, E. Fazl-Ersi, and A. Harati, "A component-based video content representation for action recognition," *Image and Vision Computing*, vol. 90, Article ID 103805, 2019.

[42] M. Bucher, S. Herbin, and F. Jurie, "Improving semantic embedding consistency by metric learning for zero-shot classiffication," in *Proceedings of the European Conference on Computer Vision*, pp. 730–746, New York, NY, USA, June 2016.

[43] Z. Al-Halah and R. Stiefelhagen, "How to transfer? Zero-shot object recognition via the hierarchical transfer of semantic attributes," in *Proceedings of the Applications of Computer Vision (WACV)*, pp. 837–843, IEEE, Waikoloa, HI, USA, January 2015.

[44] Z. Fu, T. Xiang, E. Kodirov, and S. Gong, "Zero-shot object recognition by semantic manifold distance," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2635–2644, Boston, MA, USA, June 2015.

[45] Z. Zhang and V. Saligrama, "Zero-shot learning via joint latent similarity embedding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6034–6042, Las Vegas, NV, USA, June 2016.

[46] D. Jayaraman and K. Grauman, "Zero-shot recognition with unreliable attributes," in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds., pp. 3464–3472, Curran Associates, Inc., Red Hook, NY, USA, 2014.

[47] C. H. Lampert, H. Nickisch, and S. Harmeling, "Learning to detect unseen object classes by between-class attribute transfer," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, Miami, FL, USA., June 2009.

[48] M. Norouzi, T. Mikolov, S. Bengio et al., "Zero-shot learning by a convex combination of semantic embeddings," 2014.

[49] X. Yu and Y. Aloimonos, "Attribute-based transfer learning for object categorization with zero/one training example," in *Computer Vision - ECCV 2010*, K. Daniilidis, P. Maragos, and N. Paragios, Eds., Springer, Berlin, Germany, 2010.

[50] V. Escorcia, J. C. Niebles, and B. Ghanem, "On the relationship between visual attributes and convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1256–1264, Boston, MA, USA, June 2015.

[51] J. Andreas, M. Rohrbach, T. Darrell, and D. Klein, "Neural module networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 39–48, Las Vegas, Nevada, USA, July 2016.

[52] V. Kalogeiton, P. Weinzaepfel, V. Ferrari, and C. Schmid, "Joint learning of object and action detectors," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4163–4172, Venice, Italy, October 2017.

[53] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 91–99, Montreal, Canada, December 2015.

[54] J. Liu, B. Kuipers, and S. Savarese, "Recognizing human actions by attributes," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pp. 3337–3344, Colorado Springs, CO, USA, June 2011.

[55] B. Yao, X. Jiang, A. Khosla, A. L. Lin, L. Guibas, and L. Fei-Fei, "Human action recognition by learning bases of action attributes and parts," in *Proceedings of the 2011 International Conference on Computer Vision*, pp. 1331–1338, IEEE, Barcelona, Spain, November 2011.

[56] M. Jain, J. C. Van Gemert, T. Mensink, and G. M. S. Cees, "Objects2action: classifying and localizing actions without any video example," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 4588–4596, Santiago, Chile, December 2015.

[57] M. Jain, J. C. Van Gemert, and G. M. S. Cees, "What do 15,000 object categories tell us about classifying and localizing actions?" in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 46–55, Long Beach, CA, USA, September 2015.

[58] K. Kato, L. Yin, and A. Gupta, "Compositional learning for human object interaction," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 234–251, Munich, Germany, September 2018.

[59] X. Chen, A. Shrivastava, and A. Gupta, "Neil: extracting visual knowledge from web data," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1409–1416, Sydney, Australia, July 2013.

[60] J. R. R. Uijlings, K. E. A. Van De Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *International Journal of Computer Vision*, vol. 104, no. 2, pp. 154–171, 2013.

[61] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1440–1448, Santiago, Chile, December 2015.

[62] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 779–788, Las Vegas, NV, US, June 2016.

[63] W. Liu, D. Anguelov, D. Erhan et al., "Ssd: single shot multibox detector," in *Proceedings of the European Conference on Computer Vision*, pp. 21–37, Amsterdam, The Netherlands., October 2016.

[64] R. Arandjelovic, P. Gronat, A. Torii, P. Tomas, and J. Sivic, "NetVLAD: CNN architecture for weakly supervised place recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5297–5307, Boston, MA, USA, June 2016.

[65] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3304–3311, San Francisco, CA, USA, June 2010.

[66] A. Papazoglou and V. Ferrari, "Fast object segmentation in unconstrained video," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1777–1784, Sydney, Australia, June 2013.

[67] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, “High accuracy optical flow estimation based on a theory for warping,” in *Proceedings of the European Conference on Computer Vision*, pp. 25–36, Berlin, Heidelberg, June 2004.

[68] X. Xu, T. Hospedales, and S. Gong, “Semantic embedding space for zero-shot action recognition,” in *Proceedings of the 2015 IEEE International Conference on Image Processing (ICIP)*, pp. 63–67, IEEE, Quebec City, Canada, December 2015.

[69] E. Loper and S. Bird, “Nltk: The natural language toolkit,” 2002, https://arxiv.org/abs/cs/0205028.

[70] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 3111–3119, Lake Tahoe, NV, USA, December 2013.

[71] J. Pennington, R. Socher, and C. D. Manning, “Glove: global vectors for word representation,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, Doha, Qatar, October 2014.

[72] K. Soomro, A. R. Zamir, and M. Shah, “UCF101: a dataset of 101 human action classes from videos in the wild,” 2012, https://arxiv.org/abs/1212.0402.

[73] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, “HMDB: a large video database for human motion recognition,” in *Proceedings of the 2011 International Conference on Computer Vision*, pp. 2556–2563, Barcelona, Spain, June 2011.

[74] C. Xu, S.-H. Hsieh, C. Xiong, and J. J. Corso, “Can humans fly? action understanding with multiple classes of actors,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2264–2273, Boston, MA, USA, June 2015.

[75] L. Zhang, T. Xiang, and S. Gong, “Learning a deep embedding model for zero-shot learning,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2021–2030, Honolulu, HI, USA, July 2017.

[76] I. Misra, A. Gupta, and M. Hebert, “From red wine to red tomato: composition with context,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1792–1801, Honolulu, HI, USA, July 2017.

Hindawi
Computational Intelligence and Neuroscience
Volume 2021, Article ID 6690590, 11 pages
https://doi.org/10.1155/2021/6690590

Research Article

Accurate Multilevel Classification for Wildlife Images

Francisco Gomez-Donoso, Félix Escalona, Ferran Pérez-Esteve, and Miguel Cazorla

Institute for Computer Research, University of Alicante, P.O. Box 99. 03080 Alicante, Spain

Correspondence should be addressed to Francisco Gomez-Donoso; fgomez@ua.es

Received 3 December 2020; Revised 1 March 2021; Accepted 24 March 2021; Published 2 April 2021

Academic Editor: Fabio Solari

The most common approaches for classification rely on the inference of a specific class. However, every category could be naturally organized within a taxonomic tree, from the most general concept to the specific element, and that is how human knowledge works. This representation avoids the necessity of learning roughly the same features for a range of very similar categories, and it is easier to understand and work with and provides a classification for each abstraction level. In this paper, we carry out an exhaustive study of different methods to perform multilevel classification applied to the task of classifying wild animals and plant species. Different convolutional backbones, data setups, and ensembling techniques are explored to find the model which provides the best performance. As our experimentation remarks, in order to achieve the best performance on the datasets that are arranged in a tree-like structure, the classifier must feature an EfficientNetB5 backbone with an input size of 300 × 300 px, followed by a multilevel classifier. In addition, a Multiscale Crop data augmentation process must be carried out. Finally, the accuracy of this setup is a 62% top-1 accuracy and 88% top-5 accuracy. The architecture could benefit for an accuracy boost if it is involved in an ensemble of cascade classifiers, but the computational demand is unbearable for any real application.

1. Introduction

The most common pipeline for object recognition relies in the prediction of the most specific class as stated by the dataset that is used for training the system. For instance, a dataset could be composed of the categories "pedestrian," "bike," "car," "van," and "motorbike." A classic machine learning approach using a labeled dataset would state if the input sample match any of the classes. Nonetheless, every category is inherently organized as a taxonomic tree. For instance, "bike" and "motorbike" are "two-wheel vehicles"; "car" and "van" are "four-wheel vehicles"; likewise, "two-wheel vehicles" and "two-wheel vehicles" are "vehicles," and "vehicles" and "pedestrian" are "urban objects." Thus, we organized plain categories in a taxonomic tree. The categories that are grouped under the same node share common features.

Tackling the classification problem within this framework introduces several advantages. First, the features that would learn the classifier are grouped, so it does not have to learn roughly the same features for several, slightly different categories. Then, in this case, the classifier states the proper category for each level of the taxonomic tree, so it would provide a set of predictions at different abstraction levels.

Thus, in this paper, we study different architectures, approaches, and data setups focused on performing classification on a multilevel fashion. To do so, we tackled the iNaturalist challenge, as it sets a multilevel classification problem, and being a challenge as it is, it also provides an easy comparison framework. In addition, being able to automatically recognize wildlife entities could be applied for a range of different applications. For instance, it could be used for early detection of plagues, to easily analyze the migration habits of several animals, or for protecting endangered flora and fauna population. So far, the precise identification of species is required to be performed by an expert. This takes time and effort. Thus, being able to automatically state the species of a sample could lead to take early actions that would dramatically reduce the consequences of, as we mentioned before, an insect plague in agricultural plantings.

Specifically, the main contributions of this work include the following:

(i) A study of different convolutional backbones and setups to perform multilevel classification
(ii) A study of different methods to create ensemble models for improving the accuracy of the whole system
(iii) Application of the approach to tackle the iNaturalist challenge for automatically recognizing wildlife entities in images

The rest of the document is structured as follows. First, we briefly discuss related works to classic and multilevel classification approaches and wildlife entity automatic recognition in Section 2. In Section 3, a summary of the details regarding the iNaturalist dataset are given, which is the dataset we use in our experiments. Then, in Section 4, we explain the architectures that were involved in this study, the ensemble models, and the data setup we used in the experiments. The results of the exhaustive experimentation we carried out are discussed in Section 5. Finally, the conclusions of the work and future research directions are drawn in Section 6.

2. Related Works

2.1. Image Classification. Multiclass classification of images has been a widely studied topic in the history of artificial intelligence. One of the simplest approaches to performing multiclass classification is to train a series of binary classifiers, where the output of each classifier is used to produce the final multiclass classification. This approach was explored in early neural networks and in support vector machines. The solution, although it is simple, also has serious drawbacks. For example, the feature space is not properly scanned and can lead to overfitting issues. In the past, a single classifier that made multiclass predictions was theoretically introduced. However, it could not be tested due to lack of computing power. With the emergence of massively parallel platforms such as GPUs and the deep learning paradigm, this approach has been widely explored. In [1–3], the traditional approach of a binary classifier set is compared to a pure multiclass classifier. The authors claim that the multiclass approach has several advantages, such as reduced training and inference time and a broader exploration of the feature space. However, these approaches do not take advantage of the fact that labels are organized taxonomically. In [4], it is concluded that multiclass classification works best with well-balanced data, while the approach that uses a set of classifiers works best in the presence of unbalanced data.

Since the recent increase in popularity of neural networks, the research of new network architectures has experimented a great development. The increasing number of layers in modern networks amplifies the differences between architectures and motivates the exploration of different connectivity patterns. One of the most popular architectures was the ResNet, and this kind of neural network achieved an impressive performance on many challenging image recognition tasks, such as ImageNet. However, in [5] a comparison is made with other modern architectures. Unlike ResNet architectures, DenseNet concatenates feature maps learned by different layers which increases variation in the input of subsequent layers and improves efficiency. The authors assure that compared to Inception [6] networks, which also concatenate features from different layers, DenseNet are simpler and more efficient. There are other interesting architectures that have recently emerged, one of them is EfficientNet network, and in [7], it is said that in general, the EfficientNet models achieve both higher accuracy and better efficiency over existing convolutional neural networks (CNNs), reducing parameter size, and floating point operations per second (FLOPS). Compared with the widely used ResNet50, EfficientNetB4 uses similar FLOPS and improves significantly the results on ImageNet.

2.2. Wildlife Identification. In 2017, Brust et al. [8] trained an object detection method YOLO to extract cropped images of gorilla faces. Once the faces are extracted, they trained a CNN model that had an accuracy of 90%. In this work, the authors discuss how deep learning could be really helpful for ecological studies, specifically, in the fields of identification of natural species, spatiotemporal coverage, and socio-ecological insights. Another significant work is [9], where the authors take the iNaturalist 2017 dataset as a basis to analyze the existing difficulties in identifying plants and animals. Wildlife identification belongs to a specific field of image classification in which the different categories have great visual similarities between them. This kind of identification is usually called fine-grained classification. For instance, facial identification can be seen as a type of classification with similar visual features. However, because of the underlying geometric similarity between faces, current approaches tend to perform a large amount of specific face preprocessing [10–12]. Moreover, images of natural species have their own characteristics, and for instance, individuals from the same species can differ in appearance due to sex and age and may also appear in different environments. In [13], a solution for the identification of wild animals is discussed, and in this case, the work focuses on identifying animals among 20 species from low-quality images taken from camera traps. The authors claim that unbalanced data are one of the biggest drawbacks to building an effective solution. They also pay attention to other facts, such as data augmentation or using modern network architectures as EfficientNet. However, although they use ensemble learning and admit a great improvement on these systems, it does not go too deep into them.

Due to the similarity between the different categories, only a few experts can identify accurately the corresponding class, so the number of images is usually fewer as detailed expert annotations are more difficult to obtain. To solve this problem, Cui et al. [14] proposed a data transfer learning scheme. However, this technique requires retraining models using large datasets without obtaining significant performance compared to the needed computation time.

Furthermore, as we go down into the spectrum of similarity, the number of instances in each class becomes smaller. This motivates the need for automated systems that are able to discriminate between a large number of potentially similar categories, having only a small number of examples for some categories.

2.3. Multilevel Classification. Multilevel or hierarchical classification is a very discussed problem within the machine learning community. For instance, in [15], the authors propose a cascade of classifiers to perform Attention Deficit Hyperactivity Disorder from a set of traditional features. This very same approach is used in [16] to create a generic multilevel classifier for medical datasets. Following on, different multilevel architectures are explored in the context of deep learning in [17]. The authors also applied their findings to a Diptera dataset. Finally, a comprehensive and exhaustive review on multilevel and hierarchical classification is provided in [18]. In this review, the multilevel approaches are classified in one classifier per node, which consists on a binary classifier per category; one multilabel classifier per level; and one classifier per parent node, which is also known as cascade of classifiers.

2.4. Related Datasets. There are some state-of-the-art datasets that include images of categories with great visual similarities among them. Currently, there are fine-grained datasets related with natural species, for instance, we found datasets about birds [19–21] and dogs [22, 23]. The ImageNet [24] dataset is not usually defined as a fine-grained dataset; however, it does contain several groups of fine-grained classes, including about 60 species of birds and 120 breeds of dogs. Many of these data sets were built with the intention that they would have a uniform distribution of images across the different categories. Another characteristic that this type of datasets usually has is that they only contain images of a single domain, for example, images of birds. However, they do not usually have similar categories from different domains in the same set, such as fungi, plants, insects, and reptiles.

3. The iNaturalist Dataset

To train, test, and validate the architectures, we used the iNaturalist 2019 dataset as provided by the corresponding Kaggle challenge (https://www.kaggle.com/c/inaturalist-2019-fgvc6). This dataset is composed of a high number of wildlife samples, which are labeled in a hierarchy fashion following the taxonomic rank of the biological entities. Namely, they provide the category for each level of the taxonomic tree, from first to last level. The dataset depicts images of 1010 different plants, insects, birds, and reptilians.

The iNaturalist dataset has a total of 268.243 images, each containing one of the different animal and plant species to classify. All images are labeled with the species to which each individual belongs, and in each case, we have the complete taxonomic tree of the corresponding species, as we mentioned earlier. The shape of this hierarchy is specified is as follows (from higher abstraction to the finest grain category): Kingdom, 3 categories; Phylum, 4 categories; Class, 9 categories; Order, 34 categories; Family, 57 categories; Genus, 72 categories; and Species, 1010 categories. As stated before, each image has only one category assigned for each level. Some random samples of the iNaturalist dataset are shown in Figure 1. Finally, it is worth mentioning that an exhaustive analysis of this dataset could be found in [9].

Despite providing images of fine quality and resolution, and a high number of samples, the dataset has some issues that could affect the performance of the algorithms that are trained with it.

For instance, species that share the same taxonomic categories are more similar to each other and, in some cases, can only be distinguished from some small details. For instance, in Figure 2, different species of frogs belonging to the dataset are shown. As it can be seen, distinguishing the species to which each image belongs is a difficult task that can only be done precisely by an expert, due to the high visual similarity between them. The similarity between categories increases as the categories are more fine-grain.

Another problem of this dataset that includes flora and fauna at the same time is that elements belonging to different classes appear at the same time in some photographs. In other words, insects or other animals may appear in an image labeled as vegetation. For instance, in the leftmost image of Figure 3, we can clearly observe an insect, but the labeled category is a type of plant. This kind of samples could also be challenging for the algorithms that are trained on this dataset.

In addition to that, in several occasions, the subject to be identified and the background of the image appear blurred or with a very low quality. An example of this can be seen in the central image of Figure 3, where the image has a low resolution and the bird is hard to identify. There are other cases, in which regardless of the quality of the image, the subject to identify is barely perceptible or is quite hidden in the image. In the rightmost image of Figure 3, we can see that an amphibian appears in the photograph, but only part of it is seen, since it is hidden among the grass.

Considering the features discussed here, this dataset is suitable for properly benchmarking any multilevel classification approach, also bearing in mind that some samples are extremely challenging to classify, even for a human, due to the ambiguity they represent.

4. Accurate Multilevel Classification

As explained before, our goal is to provide the best architecture to perform multilevel classification from color images. Namely, given a taxonomic classification tree, the architectures predict the most probable category for each level of the mentioned tree taking a single color image as an input. An example of the prediction provided by our architectures is shown in Figure 4.

To do this, we put to test different deep learning-based convolutional backbones. The architectures of choice were ResNet50 [25], InceptionV3 [26], DenseNet [5], and EfficientNet B5 [7]. These architectures were chosen because they are state of the art, reportedly achieving great accuracy

Figure 1: Samples of subjects present in the dataset.

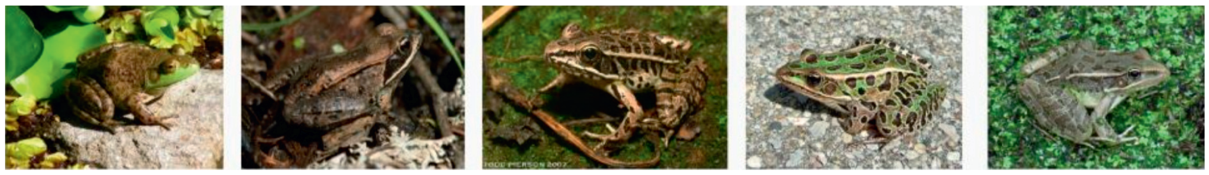


Figure 2: Similarity between species.

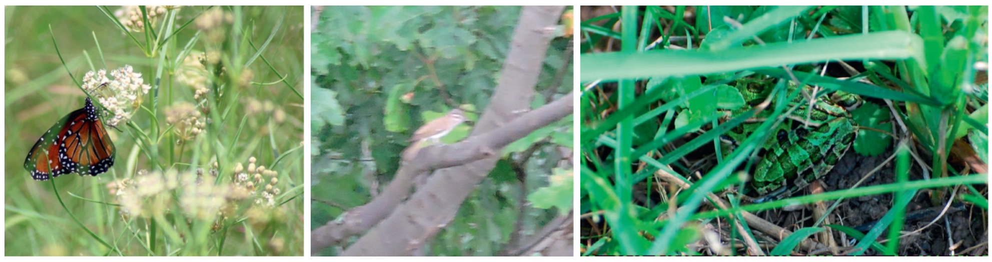


Figure 3: Samples that show challenging conditions for automatic classification algorithms.

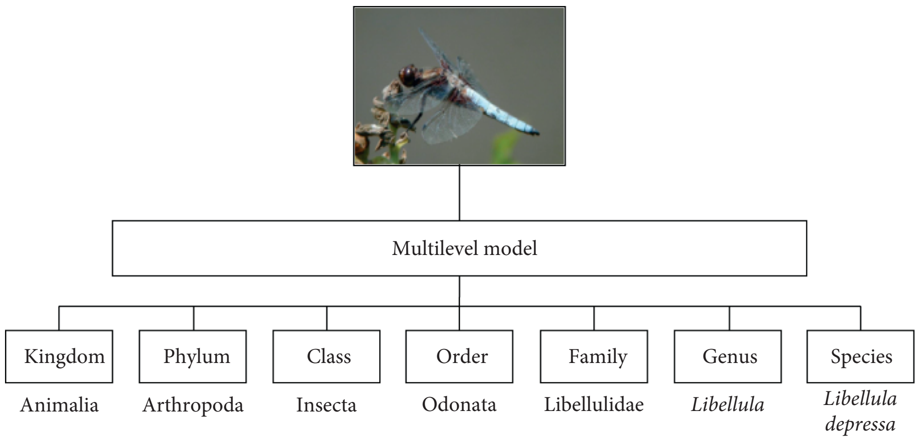


Figure 4: The proposed architectures are able to provide the most probable category for each level in a taxonomic tree.

in other datasets such as the ImageNet one. In addition, we also put to test different data setups and ensemble methodologies in order to provide the best configuration to tackle multilevel classification problems.

4.1. Classification Convolutional Backbones. As expected, the architectures mentioned before are intended for classification at the finest grain as possible, so we had to modify them in order to enable the multilevel classification. To do so, we removed the last fully connected layer of each architecture and replaced them for seven parallel fully connected layers. Namely, the feature maps provided by the convolutional backbone are forwarded to seven isolated fully connected layers. Each of these layers is a single-level classifier. We put 7 layers to match the iNaturalist annotations that provide seven levels in the taxonomic tree. The number of output neurons of each layer is different as each level has a different number of categories. For instance, the layer for the level "Species" has 1010 neurons whilst the layer for the level "Family" has 57.

It is worth noting that this methodology was applied for the experiments that involve multilevel predictions, such as the experiments explained in Section 5.3.

The experiments shown in Section 5.1 and Section 5.2 were performed following a flat classifier approach in the sake of comparison. Namely, these architecture do not perform multilevel classification, but they provide directly the finest-grain label as possible. To to this, the convolutional backbone is connected to a single fully connected layer with 1010 neurons, matching the number of categories of the "Species" level, which is the most specific level provided by the dataset, as mentioned before. In all cases, the activation function of the output neurons is softmax, which is defined as in equation (1). This function is applied to generate a probability for each class from the logits provided by the convolutional backbone:

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}. \tag{1}$$

Whether the benchmarked approach was multilevel or flat, all the architectures involved in the experiments were trained following the same setup. First, the dataset already provides a train set and a test set, so we adopted the splits with no modifications. Some experiments involved different data augmentation techniques, which we applied to the training set as explained in Section 4.2. In each training procedure, the data are shuffled so they are fed to the network with a random order. The optimization procedure was carried out by the Adam solver which was initialized with a learning rate of 0.0001. The Adam weight update protocol is shown in equation (2). The error function to minimize was categorical crossentropy. The architectures were initially configured to be trained on a forever loop, and an early stopping criterium was used to halt the training procedure. This criterium consisted of 10 consecutive tests with no significant improvement in the classification accuracy on the test set. A test was performed every 10 training epochs:

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1-\beta_1) g_t, \\ v_t &= \beta_2 v_{t-1} + (1-\beta_1) g_t^2, \\ \widehat{v}_t &= \max(\widehat{v}_{t/1}, v_t), \\ \theta_{t+1} &= \theta_t - \frac{\eta}{\sqrt{\widehat{v}_t} + \varepsilon} m_t. \end{aligned} \tag{2}$$

4.2. Data Setup. Regarding the data, the architectures were trained on the training split and tested with the test split. Both sets are provided by the dataset itself, and we adopted them with no modifications. The original training split is composed of 187770 samples, whilst the test split has 80529 samples. It is worth bearing in mind that the images of the dataset are of different resolution, so they are resized to fit the input size of each architecture. This original data setup was explored in the experiments and is referred as the "No" data augmentation setting.

Some of the experiments we carried out involve different data augmentation techniques. The data augmentation is a common method to artificially create new samples from the existing ones by slightly modifying them. This method is a default procedure when it comes to train machine learning methods and specially to train deep learning models, which is reportedly used to improve its generalization capabilities. We applied data augmentation on the least represented categories in order to match the number of samples of the most represented category of the dataset.

First, the Standard data augmentation technique consists on the application of a range of different operators. Namely, horizontal and vertical flipping, color channel shifting, Gaussian noise addition, brightness and contrast alterations, and Gaussian blur were randomly combined and applied, with random parameters each. The Standard data augmentation is applied to the training data, so an augmented training set of 353500 samples is generated. This set is the same for all the experiments that involved the Standard data augmentation technique.

Then, the Central Crop data augmentation technique consists on generating new samples by extracting the central patch of each image. The center patch covered over the 80% of the original image. As the images of the dataset are of different resolution, the crop is performed before the resize operation. This data augmentation technique is applied together with the Standard setup described before, so when an experiment is setup with the Central Crop method, it also includes the Standard one. We applied this technique because sometimes the input image depicts too much background in addition to the labeled sample. By cropping the central patch, we try to remove the uninteresting background whilst keeping the visual features of the target sample intact.

Finally, the Multiscale Crop data augmentation technique is about extracting the central crop of the image but at different resolution. In this case, three central crops of random size, ranging from 80% to 50% the size of the original sample, were extracted following the same methodology explained before. In this case, the Standard data augmentation method is also applied alongside the Multiscale one. The reason behind the application of this method is that the relative size of the target sample with respect to the whole image is not fixed. Thus, we can find samples where the subject covers the whole image, and others where it is depicted on a small section of it, and the rest is background.

Some samples after applying these data augmentation techniques are shown in Figure 5.

4.3. Ensemble Methodologies. A common technique to improve the results of a classification system is to combine the decision of different classifiers to provide the final result. The goal of this method is to enhance the strong points while complementing the weakest features of each classifier. We

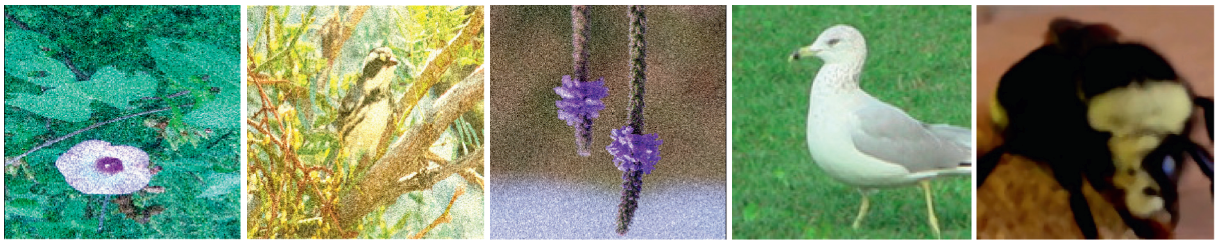


Figure 5: Some random augmented samples.

implemented three different ensemble models: boosting, stacking, and a cascade of classifiers which is a collection of specialized classifiers. These ensemble methodologies are explored in the experiments shown in Section 5.4.

Boosting [27] is based on weighting the decision of each architecture based on its theoretical accuracy on the validation set as described in equation (3). In our case, different architectures are fed so the individual decisions are computed. Then, the results of each architecture are multiplied by its theoretical accuracy. Next, the results of all the architectures are summed together and the highest scores for each level are returned as the final decision of the ensemble:

$$\widehat{d(x)} = \sum y_i \cdot w_i. \tag{3}$$

Regarding the stacking technique [28], it is based on creating an intelligent system that learns the mistakes that make a range of other classifiers, which outputs are used as the input, in order to correct them. In our case, we trained a fully connected layer that takes the output of a range of different multilevel architectures as an input and predicts the final multilevel decision.

Finally, the specialized cascading classifiers [29] consist of creating specialized classifiers for each possible category in a multilevel approach. Thus, the prediction of a specific level is trusted to be correct in order to forward the data to the proper specific classifier which will provide the prediction for the next level, namely, the next classifier. As we have 9 levels deep and more than 1000 categories, training one classifier for each is unbearable. Thus, we only performed this technique for the Phylum level, which includes 4 different categories. Thus, in our incarnation of this approach, a classifier is in charge of deciding the proper Phylum of the input sample. Then, based on this decision, the sample is forwarded to the classifier of the predicted Phylum so it states the rest of the more specific levels. Namely, we trained a general classifier that predicts the Phylum, and a specialized classifier for each Phylum that predicts the category for the rest of the levels.

5. Experimentation and Discussion

In this section, the results of the experiments we carried out are reported. In addition to each setup, we also provide the top-1 and top-5 accuracy.

All the experiments were conducted on the same machine, which features an Intel i7-7700 @ 3.6 GHz CPU with 16 GB of DDR3 RAM. The chipset of choice is Z370. All the computations were accelerated by a Nvidia Quadro P6000 GPU unit. Regarding the software, the SO is Ubuntu 18.04 LTS and all the architectures were implemented on Keras 2.2.4 and tensorflow 1.4.0 frameworks using CUDA 9.0 and CuDNN.

5.1. Baseline Convolutional Backbone Flat Classifiers. In this section, the experiments are intended to compare different convolutional backbones. In addition to that, it is also intended to discover whether the data augmentation impacts on the accuracy. It is worth noting that these experiments do not involve multilevel classification, but just flat classification on the most specific level of the taxonomy. The results are reported in Table 1.

As it can be seen, the most accurate convolutional architectures are the DenseNet201 and the EfficientNetB5 with a marginal accuracy difference between them. On one hand, the DenseNet architecture features residual connections between every convolutional block. This fact allows the network to explicitly learn powerful multiscale features that are very helpful for correctly classifying the images in which the sample is depicted in a small part. In a convolutional pipeline with no residual connections, the mentioned visual features would disappear as per effect of the convolution and pooling operations in the earliest stages of the network. On the other hand, the EfficientNet family of architectures was created by performing a Neural Architecture Search. Namely, this collection of architectures was proposed by an intelligent system that aims to achieve the best trade-off between accuracy and runtime. The EfficientNetB5 is the most powerful in terms of accuracy, and it provides the best top-1 and top-5 results too in our experiments.

Regarding the inclusion of data augmentation, it clearly impacted the accuracy of every architecture. As it can be seen, all of them experienced a significant accuracy boost when data augmentation was involved in both top-1 and top-5 metrics.

Thus, we can conclude that the setups involving data augmentation techniques, and DenseNet101 and EfficientNetB5 convolutional backbones provide the best accuracy.

5.2. Resolution Experiments on Flat Classifiers. In this section, we put to test the impact of the input resolution on the accuracy of the architectures for flat classification. No multilevel prediction is involved in these experiments. Despite being DenseNet101 and EfficientNetB5, the architectures that provided the best performance on the

Table 1: Results on flat classification for different convolutional backbones and data augmentation.

ID	Conv. backbone	Resolution	Data augmentation	Top-1	Top-5
BA	ResNet50	250×250	No	0.27	0.62
BB	ResNet50	250×250	Standard	0.45	0.73
BC	InceptionV3	250×250	No	0.34	0.63
BD	InceptionV3	250×250	Standard	0.46	0.75
BE	DenseNet201	250×250	No	0.33	0.64
BF	DenseNet201	250×250	Standard	0.51	0.77
BG	EfficientNetB5	250×250	No	0.36	0.65
BH	EfficientNetB5	250×250	Standard	0.53	0.8

experiments of Section 5.1, they are very computationally expensive. Namely, due to hardware restrictions, we cannot test these networks with a resolution greater than 300×300 pixels. Thus, we adopted ResNet50 to carry out these experiments up to 350×350 and we will assume that the results apply to DenseNet and EfficientNetB5.

The results of testing ResNet50 with a range of different resolutions are shown in Table 2. For this particular problem, the accuracy is proportional to the resolution. Namely, the accuracy improves as the resolution of the input images increases. It is expected that the improvement becomes marginal at a certain resolution onwards, but we cannot reach that limit due to hardware limitations as explained before. It is also worth noting that the accuracy of ResNet50 with a resolution of 350×350 is similar to the accuracy achieved by EfficientNetB5 with 250×250 resolution, being also similarly computationally expensive.

At this point, the setup that provides the best accuracy is DenseNet and EfficientNet with data augmentation and the largest resolution allowed by the hardware, which in our case is 300×300 pixels.

5.3. Convolutional Multilevel Classifiers. In this section, we benchmarked the best setup for performing multilevel classification. As the experiments we performed before on flat classification conclude, only DenseNet and EfficientNetB5 are considered because they provide the best accuracy. In addition, 250×250 and 300×300 resolutions are put to test. Finally, the three different types of data augmentation explained in Section 4.2 are also benchmarked.

The results of the experiments on multilevel classification are reported in Tables 3 and 4. Regarding the convolutional backbone, the one that provides the best top-1 and top-5 accuracy regardless the input image resolution and the data augmentation type is EfficientNetB5. EfficientNetB5 consistently outperformed DenseNet201 in every experiment if it is compared pairwise with the same setup.

The impact of the resolution is also noticeable in this experiment, as the increment of resolution led to an accuracy boost in every experiment as well. This is expected as the visual features of the samples are of more quality, and they also can reach deeper in the convolutional pipeline, which improves the classification accuracy. This effect is more noticeable on the DenseNet101 architecture as it explicitly provides multiscale feature extraction through its dense residual connections.

The inclusion of Central Crop and Multiscale Crop data augmentation techniques increased the accuracy of the models too. However, the improvement is marginal respect to the Standard method. It seems that it is preferable to enlarge the input resolution rather than to apply Central Crop or Multiscale Crop data augmentation techniques when possible. In addition, we can conclude that the Standard data augmentation technique indeed helps to improve the generalization capabilities and the overall accuracy of the model in this multilevel classification configuration.

As we can see in Figure 6, the probability distributions computed with the F1-score results show that the performance of both models look pretty similar. F1-scores for the ML model can be represented as a normal distribution of 0.60 mean and 0.04 of typical deviation, while the results for MF model can be expressed as a normal distribution of 0.62 mean and 0.03 of typical deviation.

These distributions mean that the ML model has approximately 659 classes (65.2%) whose F1-score is between 0.56 and 0.64, and 964 classes (95.4%) whose F1-score varies from 0.52 to 0.68. On the other hand, the MF model has approximately 659 classes (65.2%) whose F1-score is between 0.59 and 0.65, and 964 classes (95.4%) whose F1-score varies from 0.56 to 0.68.

In the light of the experiments, both models (DenseNet201 and EfficientNetB5 with Multiscale Crop) show a similar performance, DenseNet201 is slightly superior in the F1-score and AUC score test, whilst EfficientNetB5 is moderately superior in the top-1 and top-5 test, which we considerate crucial, so we selected EfficientNetB5 as the convolutional backbone for performing multilevel classification of wildlife imagery, with input images of 300×300 pixels resolution and Multiscale Crop data augmentation. However, it is worth mentioning that the improvement over the Standard data augmentation configuration is marginal.

5.4. Multilevel Ensemble Approaches. In this section, we put to test the performance on multilevel classification of the ensemble setups discussed on Section 4.3. As the aim of the ensemble models is to combine different approaches so that their weaknesses are compensated and their strong points enhanced, we chose the setups MF and ML, as reported in Table 3, to create the ensembles. These setups provide the best accuracy levels thus far and feature two different convolutional backbones.

The results of both ensemble methods, boosting and stacking, are reported in Table 5. In the light of the results, both methods led to an important improvement in terms of accuracy compared with using just one architecture. Furthermore, the boosting approach provides an even better accuracy, reaching a top-1 71%.

We also tested the cascade of classifiers approach. Despite a pure cascade of classifiers would involve a classifier per category in each level, we only trained specialized

Table 2: Results of flat classification for a range of different input image resolutions.

ID	Conv. backbone	Resolution	Data augmentation	Top-1	Top-5
RA	ResNet50	200 × 200	Standard	0.43	0.72
RB	ResNet50	250 × 250	Standard	0.45	0.73
RC	ResNet50	300 × 300	Standard	0.48	0.76
RD	ResNet50	350 × 350	Standard	0.53	0.83

Table 3: Results for multilevel classification that includes different convolutional backbones, input image resolution, and data augmentation techniques.

ID	Conv. backbone	Resolution	Data augmentation	Top-1	Top-5
MA	DenseNet201	250 × 250	Standard	0.51	0.78
MB	DenseNet201	250 × 250	Central Crop	0.52	0.79
MC	DenseNet201	250 × 250	Multiscale Crop	0.51	0.78
MD	DenseNet201	300 × 300	Standard	0.58	0.83
ME	DenseNet201	300 × 300	Central Crop	0.59	0.84
MF	DenseNet201	300 × 300	Multiscale Crop	0.61	0.85
MG	EfficientNetB5	250 × 250	Standard	0.53	0.82
MH	EfficientNetB5	250 × 250	Central Crop	0.53	0.81
MI	EfficientNetB5	250 × 250	Multiscale Crop	0.54	0.84
MJ	EfficientNetB5	300 × 300	Standard	0.61	0.87
MK	EfficientNetB5	300 × 300	Central Crop	0.62	0.87
ML	EfficientNetB5	300 × 300	Multiscale Crop	0.62	0.88

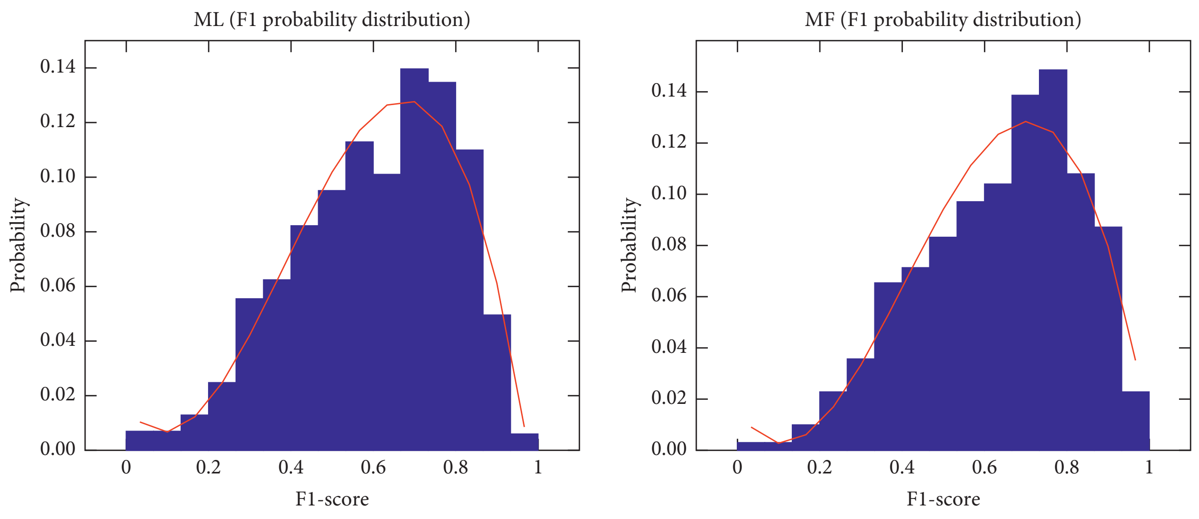


Figure 6: Probability distribution of F1-score for the ML (leftmost) and MF (rightmost) experiments.

Table 4: Results for multilevel classification that includes area under the ROC curve and weighted F1-score.

ID	Conv. backbone	Resolution	Data augmentation	AUC	F1-score
MF	DenseNet201	300 × 300	Multiscale Crop	0.812	0.62
ML	EfficientNetB5	300 × 300	Multiscale Crop	0.806	0.60

classifiers for each Phylum in this case. As we have 9 levels deep and more than 1000 categories, training one classifier for each is unbearable. The setup of choice for the classifiers was based on the experiment ML, as reported in Table 3. The results for this setup show a slightly better accuracy compared with the original ML experiment. Nonetheless, even if the improvement was not marginal, the requirement of a single classifier per category in each level makes this approach very time consuming, which could be inadequate for a range of applications.

Finally, some results of correctly classified subjects as provided by the EC setup are shown in Figure 7. As it can be seen, the approach is able to work with images that do not depict the subject entirely, with cluttered scenarios, with several subjects in the same picture and with a range of different poses.

In addition, some errors are shown in Figure 8. In this case, the first image depicts a sample of *Thamnophis sirtalis* and the second one a sample of *Thamnophis elegans* incorrectly classified. The main difference

Table 5: Results for multilevel ensemble methods.

ID	Setup	Ensemble Mode	Top-1	Top-5
EA	MF + ML	Boosting	0.71	0.95
EB	MF + ML	Stacking	0.68	0.93
EC	ML	Cascade classifiers	0.63	0.89

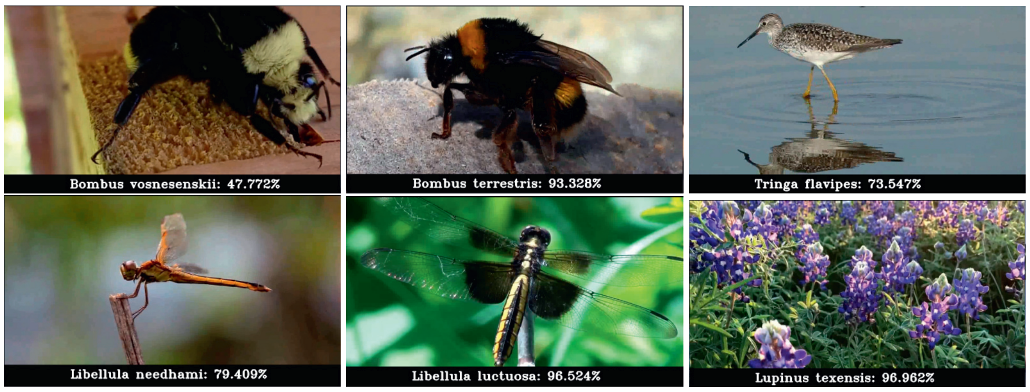

Bombus vosnesenskii: 47.772%
Bombus terrestris: 93.328%
Tringa flavipes: 73.547%
Libellula needhami: 79.409%
Libellula luctuosa: 96.524%
Lupinus texensis: 96.962%

Figure 7: Random samples correctly classified as provided by the setup EC.

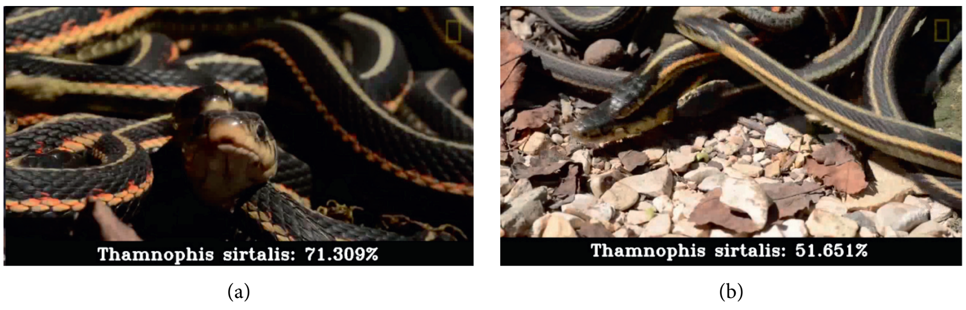


Figure 8: A sample of *Thamnophis sirtalis* (correctly classified) and sample of *Thamnophis elegans* (incorrectly classified). Note the high visual similarity of both species.

between these two species is that the first has a dash-like pattern of red marks along the lateral parts of its body. This kind of error happens sometimes with species that look so similar.

6. Conclusion and Future Work

In this work, we have presented an exhaustive study of different methods to perform multilevel classification from color images applied to the problem of classifying wild animals and plant species.

In order to solve this problem, data from different competitions of *iNaturalist* have been fused and processed with data augmentation techniques. Moreover, several different state-of-the-art architectures have been adapted to the taxonomy of our problem to find the best model.

Our experiments show that increasing the resolution of the images impact on the final accuracy, as the finer details are very important to determine the exact specie of each being are preserved. In addition to that, the best architecture is EfficientNetB5 with the Multiscale data augmentation method. Furthermore, the ensemble models show even better accuracy, being the boosting technique that provides the best results.

It is important to note that, additionally to the results presented in this paper, our system has been applied with success to a range of real-life videos that contain moving species. A sample of this functionality could be seen at https://youtu.be/rzEIYt4Gj0A. This test has let us check the performance of our system in real scenarios, being capable of classifying more than 1000 species with high robustness.

Regarding the limitation of the methods, they still have room for improvement in terms of accuracy. In addition to that, it is worth noting that the error of the classifiers is greater as they got deeper in the taxonomy. Namely, the classifier of the Species level is the most prone to error. This is expected as the samples of different categories are more visually alike. Thus, in future research studies, we plan to tackle this problem so that the accuracy is kept no matter the depth level in the taxonomic tree.

Data Availability

The data used to support the findings of this study are currently under embargo while the research findings are commercialized. Requests for data, (6/12 months) after publication of this article, will be considered by the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was funded by the Spanish Government PID2019-104818RB-I00 grant, supported with FEDER funds. It was supported by Spanish grants for PhD studies ACIF/2017/243 and FPU16/00887. The authors would like to thank Nvidia (Santa Clara, California, USA) for the generous donation of a Titan Xp and a Quadro P6000.

References

[1] E. Allwein, R. Schapire, and Y. Singer, "Reducing multiclass to binary: a unifying approach for margin classifiers," *Journal of Machine Learning Research*, vol. 1, pp. 113–141, 2000.

[2] C. W. Hsu and C. J. Lin, "A comparison of methods for multiclass support vector machines," *IEEE Transactions on Neural Networks*, vol. 13, no. 2, pp. 415–425, 2002.

[3] G. OU and Y. L. Murphey, "Multi-class pattern classification using neural networks," *Pattern Recognition*, vol. 40, no. 1, pp. 4–18, 2007.

[4] R. Babbar, I. Partalas, E. Gaussier, and M.-R. Amini, "On flat versus hierarchical classification in large-scale taxonomies," in *Proceedings of the 27th Annual Conference on Neural Information Processing Systems (NIPS 26)*, pp. 1824–1832, Lake Tahoe, NV, USA, December 2013.

[5] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, July 2017.

[6] C. Szegedy, V. Vanhoucke, J. Shlens, and Z. W. S. Ioffe, "Rethinking the inception architecture for computer vision," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, June 2016.

[7] M. Tan and Q. V. Le, "Efficientnet: rethinking model scaling for convolutional neural networks," 2019, https://arxiv.org/abs/1905.11946.

[8] C.-A. Brust, T. Burghardt, M. Groenenber et al., "Towards automated visual monitoring of individual gorillas in the wild," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2820–2830, Honolulu, HI, USA, July 2017.

[9] G. V. Horn, O. M. Aodha, Y. Song et al., "The iNaturalist species classification and detection dataset," *The Quarterly Journal of Experimental Psychology*, vol. 18, no. 4, pp. 362–365, 2018.

[10] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," in *Proceedings of the British Machine Vision Conference*, Swansea, UK, September 2015.

[11] F. Schroff, D. Kalenichenko, and J. P.. Facenet, "A unified embedding for face recognition and clustering," in *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, USA, June 2015.

[12] Y. Taigman, M. R. M. Yang, and L. Wolf, "Deepface: closing the gap to human-level performance in face verification," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Columbus, OH, USA, June 2014.

[13] A. Abuduweili, X. Wu, and X. Tao, "Efficient method for categorize animals in the wild," 2019.

[14] S. Cui, Y. Song, Y. Sun, C. Howard, and A. Belongie, "Large scale fine-grained categorization and domain-specific transfer learning," in *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, June 2018.

[15] S. Itani, F. Lecron, and P. Fortemps, "A multi-level classification framework for multi-site medical data: application to the ADHD-200 collection," *Expert Systems with Applications*, vol. 91, pp. 36–45, 2018.

[16] M. Srinivas, R. Bharath, P. Rajalakshmi, and C. K. Mohan, "Multi-level classification: a generic classification method for medical datasets," in *Proceedings of the 2015 17th International Conference on E-health Networking, Application Services (HealthCom)*, pp. 262–267, Boston, MA, USA, October 2015.

[17] J. Navarrete, F. Gomez-Donoso, D. Viejo, and M. Cazorla, "Multilevel classification using a taxonomy applied to recognizing diptera images," in *Proceedings of the 2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, Budapest, Hungary, July 2019.

[18] C. N. Silla Jr. and A. A. Freitas, "A survey of hierarchical classification across different application domains," *Data Mining and Knowledge Discovery*, vol. 22, no. 1-2, pp. 31–72, 2011.

[19] T. Berg, J. Liu, S. W. Lee, M. L. Alexander, D. W. Jacobs, and P. N. B.. Birdsnap, "Large-scale fine-grained visual categorization of birds," in *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, Columbus, OH, USA, June 2014.

[20] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The Caltech-UCSD birds-200-2011 dataset," Technical Report CNS-TR-2011-001, California Institute of Technology, Pasadena, CA, USA, 2011.

[21] P. Welinder, S. Branson, T. Mita et al., "Caltech-UCSD birds 200," Technical Report CNS-TR-2010-001, California Institute of Technology, Pasadena, CA, USA, 2010.

[22] A. Khosla, N. Jayadevaprakash, B. Yao, and L. Fei-Feir, "Novel dataset for fine-grained image categorization," in *Proceedings of the FGVC Workshop at CVPR*, Salt Lake City, UT, USA, June 2011.

[23] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. V. Jawahar, "Cats and dogs," in *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3498–3505, IEEE, Providence, RI, USA, June 2012.

[24] O. Russakovsky, J. Deng, H. Su et al., "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015, https://arxiv.org/abs/1512.03385.

[26] C. Szegedy, V. Vincent, Sergey Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," 2015, https://arxiv.org/abs/1512.00567.

[27] A. Lemmens and C. Croux, "Bagging and boosting classification trees to predict churn," *Journal of Marketing Research*, vol. 43, no. 2, pp. 276–286, 2006.

[28] S. Džeroski and B. Ženko, "Is combining classifiers with stacking better than selecting the best one?" *Machine Learning*, vol. 54, no. 3, pp. 255–273, 2004.

[29] V. Paul and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) 2001*, vol. 1, IEEE, Kauai, HI, USA, April 2001.