CHAPTER 16

Generative adversarial network for video anomaly detection

Thittaporn Ganokratanaa[a] and Supavadee Aramvith[b]
[a]Department of Electrical Engineering, Faculty of Engineering, Chulalongkorn University, Bangkok, Thailand
[b]Multimedia Data Analytics and Processing Research Unit, Department of Electrical Engineering, Faculty of Engineering, Chulalongkorn University, Bangkok, Thailand

16.1 Introduction

Video anomaly detection (VAD) has gained increasing recognition in a surveillance system for ensuring security. VAD is a challenging task due to the high appearance structure of the images with motion between frames. This anomaly research has drawn interests from researchers in computer vision areas. The traditional approaches including the social force model (SF) [1], mixture of probabilistic principal component analyzers (MPPCA) [2], Gaussian mixture of dynamic texture (MDT) and the combination of SF + MPPCA [3, 4], sparse reconstruction [5–7], one class learning machine [8], K-nearest neighbor [9], and tracklet analysis [10, 11] are proposed to challenge the anomaly detection problem due to their performance in detecting multiple objects. However, the traditional approaches do not perform well with anomaly detection problem since this problem is a complex task that mostly occurs in the crowded scenes, making it more problematic for the traditional approaches to generalize. Thus, deep learning approaches are employed to achieve a higher anomaly detection rate, such as deep Gaussian mixture model (GMM) [12, 13], autoencoder [14, 15], and deep pretrain convolution neural network (CNN) [16–19]. Even using these deep learning approaches, the problem is still open when dealing with all issues of anomaly detection. Specifically, the major challenging issues in the anomaly detection task are categorized into three types: complex scene, small anomaly samples, and object localization in pixel level. With the complex scene, one may consist of multiple moving objects with clutter and occlusions that cause the difficulty in detecting and localizing objects. This issue also refers to the crowded scene which is more challenging than the uncrowded scene. The second challenge is the small samples from available anomaly datasets with abnormal ground truth, leading to the struggles of model training in a data-hungry deep learning approach. In practice, it is impossible to train all anomalous events as they occur randomly. Therefore, the anomaly detection task is categorized as an unsupervised learning manner since there are no requirements of data labeling on the positive rare class. Another important issue is about

Generative Adversarial Networks for Image-to-Image Translation
https://doi.org/10.1016/B978-0-12-823519-5.00011-7

pixel localization of the objects in the scene. Previous works [1, 6, 15, 20] struggled with this challenging task in which they achieved high accuracy only at frame-level anomaly detection. On the other hand, the accuracy of a pixel-level anomaly localization is significantly poorer. In recent works [21–23] researchers try to improve the performance to cover all evaluation criteria but can achieve good performance either at the frame or pixel level in some complex scenes. This happens as a consequence of insufficient input features for training the model such as appearance and motion patterns of the objects. The features of foreground objects should be extracted sufficiently and efficiently during training to make the model understand all characteristics.

To deal with these challenges, the unsupervised deep learning-based approach is the most suitable technique for the anomaly detection problem since it does not require any labeled data on abnormalities. Unsupervised learning is a key domain of deep generative models, such as adversarially trained autoencoders (AAE) [24], variational autoencoder (VAE) [25], and generative adversarial networks (GAN) [26, 27]. Generative models for anomaly detection aim to model only normal events in training as it is the majority of the patterns. The abnormal events can be distinguished by evaluating the distance from the learned normal events. Early generative works are mostly based on handcrafted features [1, 3, 5, 11, 28] or CNN [17, 18] to extract and learn the important features. However, the performances of anomaly detection and localization are still needed to be improved due to the difficulties in approximating many probabilistic computations and utilizing the piecewise linear units as in the generative models [29–31]. Hence, recent trends for video anomaly detection focus more on GAN [20–22] which is an effective approach that achieves high performance in image generation and synthesis, affords data augmentation, and overcomes classification problems in the complex scenarios.

16.1.1 Anomaly detection for surveillance videos

Video surveillance has gained increasing popularity since it has been widely used to ensure security. Closed-circuit television (CCTV) cameras are used to monitor the scene, record certain situations, and provide evidence. They generally perform as the postvideo forensic process that allows the investigation for abnormalities of previous events in manual control by human operators [32]. This manual causes difficulty for the operators since abnormalities can occur in any situation, such as crowded or uncrowded in indoor and outdoor scenes. Additionally, it may cause serious problems, including a terrorist attack, a robbery, and an area invasion, leading to personal injury or death and property damage [33]. Thus, to enhance the performance of video surveillance, it is crucial to building an intelligent system for anomaly detection and localization.

The anomaly is defined as "a person or thing that is different from what is usual, or not in agreement with something else and therefore not satisfactory" [34]. Multiple terms

Fig. 16.1 Examples of abnormal events in crowds from UCSD pedestrian [4], UMN [1], and CHUK Avenue datasets [6].

stand for the anomaly, including anomalous events, abnormal events, unusual events, abnormality, irregularity, and suspicious activity. In VAD, the abnormal event can be seen as the distinctive patterns or motions that are different from neighboring areas or the majority of the activities in the scene. Specifically, the normal events are the frequently occurring objects and the common moving patterns representing the majority of the patterns, while the abnormal events are varied and rarely occurred describing as an infrequent event that may include an unseen object and have a significantly lower probability than the probability of the normal event. Examples of different abnormal events are shown in Fig. 16.1.

The anomaly detection for surveillance videos is challenging because of complex patterns of the real scene (e.g., moving foreground objects with large amounts of occlusion and clutter in crowds) captured by the static CCTV cameras. The VAD relies on fixed CCTV cameras, which take only moving foreground objects into account while disregarding the static background. The goal of VAD is to accurately identify all possible anomalous events from the regular normal patterns in crowded and complex scenes from the video sequences. To design the effective anomaly detection for surveillance videos, it is considered to learn all information of the objects from both of their appearance (spatial) and motion (temporal) features under the unsupervised learning or semisupervised learning manner. In the model training, with the unsupervised learning task, only the frames of normal events are trained, meaning that there is no data labeling on abnormalities. This benefits the use of VAD in real-world environments where any type of abnormal events can unpredictably occur. Then, all videos are fed into the model during testing. Any pattern deviated from the trained normal samples is identified as abnormal events that can be detected by evaluating the anomaly score known as the error of the predictive model in a vector space or the posterior probability of the test samples.

16.1.2 A broader view of generative adversarial network for anomaly detection in videos

A GAN has been studied for years. The success of GAN comes from the effectiveness of its structure in improving image generation and classification tasks with a pair of networks. GAN presents an end-to-end deep learning framework in modeling the

likelihood of normal events in videos and provides flexibility in model training since it does not require annotated abnormal samples. Its learning is achieved through deriving backpropagation to compute the error of each parameter in both generator and discriminator networks. The goals of GAN are to produce the synthetic output that is not able to be identified as different from the real data and to automatically learn a loss function to achieve the indistinguishable output goal. The loss of GAN attempts to classify whether the synthetic output is fake or real, while it is trained to be minimized in the generative model at the same time. This loss makes it more beneficial to apply GAN in various applications since it can adapt to the data without requiring different loss functions, unlike the loss functions of the traditional CNNs approach.

Specifically, in the video anomaly detection, GAN performs as a two-player mini-max game between a generator G and a discriminator D, providing high accuracy output. G attempts to fool D by generating synthetic images that are similar to the real data, whereas D efforts to discriminate whether the synthetic image belongs to the real or synthetic data. This mini-max game benefits data augmentation and implicit data management, thanks to D that assists G to reduce the distance between its samples and the training data distribution and to train on the small benchmark without the need to define an explicit parametric function or additional classifiers. Therefore, GAN is one of the most distinctive approaches to deal with complex anomaly detection tasks since it achieves good results in reconstructing, translating, and classifying images. Following the unsupervised GAN in an image-to-image translation [35], it can extract significant features of the objects of interest (e.g., moving foreground objects) and efficiently translate them from spatial to temporal representations without any prior knowledge of anomaly or direct information on anomaly types. In this way, GAN can provide comprehensive information concerning appearance and motion features. Hence, we focus on reviewing GAN for the anomaly detection task in videos and also introduce our proposed method named deep spatiotemporal translation network or DSTN, as a novel unsupervised GAN approach to detect and localize anomalies in crowded scenes [26].

This chapter contains six sections. In Section 16.1, anomaly detection for surveillance videos and a broader view of GAN are reviewed. Section 16.2 presents a literature review, including the basic structure of GAN along with the literature of anomaly detection in videos based on GAN. We elaborate on the GAN training in Section 16.3, which includes the image-to-image translation and our proposed DSTN. The performance of DSTN is discussed in Section 16.4 along with its related details, including the publicly available anomaly benchmarks, the evaluation criteria, the comparison of GAN with an autoencoder, and the advantages and limitations of GAN for anomaly detection in videos. Finally, Section 16.5 provides a conclusion for this chapter.

16.2 Literature review

We introduce the basic structure of the GAN and review the related works on anomaly detection for surveillance videos based on GAN. The details of GAN architecture and its state-of-the-art methods in video anomaly detection are described as following.

16.2.1 The basic structure of generative adversarial network

Since the concept of generative models has been studied in machine learning areas for many years, it has gained wide recognition from Goodfellow et al. [27] who introduced a novel adversarial process named GAN. The basic structure of GAN consists of two networks working simultaneously against each other, the generator G and the discriminator D as shown in Fig. 16.2. In general, G produces a synthetic image n from the input noise z, whereas D attempts to differentiate between n and a real image r. The goals of G are to generate synthesized examples of objects that look like the real ones and then attempt to fool D to make a wrong decision that the synthesized data generated by G are real. On the other hand, D has been learned on a dataset with the label of the images. D tries its best to discriminate whether its input data are fake or real by comparing them with the real training data. In other words, G is a counterfeiter producing fake checks, while D is an officer trying to catch G. Specifically, G is good at creating the synthesized images as it only updates the gradient through D to optimize its parameters, making D more challenging to differentiate its input data. The training of this mini-max game makes both networks better until at one point that the probability distributions of G and the real data are

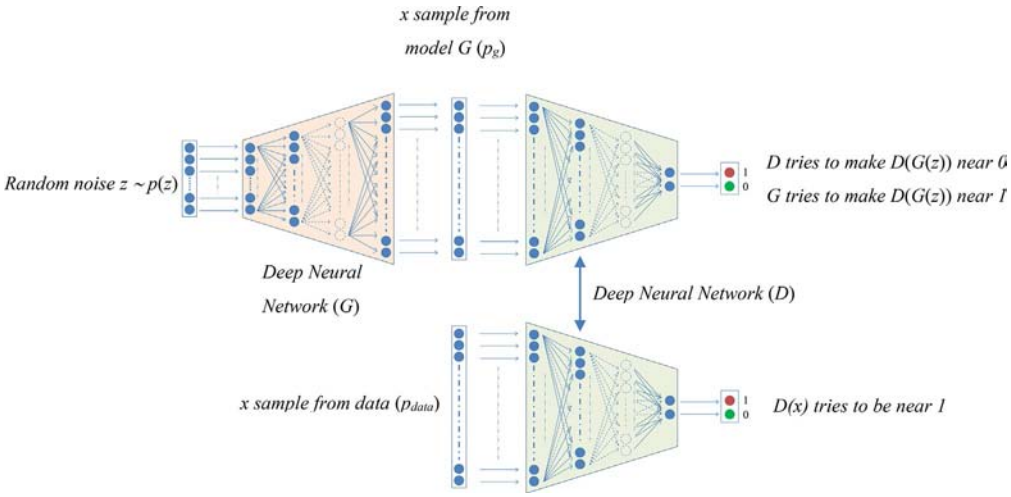

Fig. 16.2 Generative adversarial network architecture.

equivalent (when there is enough capacity and training time) so that G and D are not able to improve any longer. Thus, D is unable to differentiate between these two distributions.

In the perspective of the generative adversarial network training, G takes the input noise z from a probability distribution$p_z(z)$ and then it generates the fake data and feeds into D as $D(G(z))$. $D(x)$ denotes the probability that x is from the distribution of real datap_{data} instead of the distribution of generator p_g. The discriminator D takes two inputs from $G(z)$ and p_{data}. D is trained to enlarge the probability of defining the precise label to the real and the synthesized examples. Specifically, the goal of D is to accurately classify its input samples by giving a label of 1 to real samples and a label of 0 to synthetic ones. D tries to solve a binary classification problem based on neuron networks with a sigmoid function, giving output in the range [0,1]. Then G is simultaneously trained to reduce $[\log(1 - D(G(z)))]$. These two adversarial networks, G and D, are represented with value function $V(G, D)$ as follows:

$$\min_G \max_D V(D, G) \tag{16.1}$$

$$V(D, G) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))], \tag{16.2}$$

where $[\log D(x)]$ is the objective function of the discriminator, representing the entropy of the real data distribution p_{data} passing through D, which tries to maximize $[\log D(x)]$. Note that the objective function of the discriminator will be maximized when both real and synthetic samples are accurately classified to 1 and 0, respectively. The objective function of the generator is $[\log(1 - D(G(z)))]$, representing the entropy of the random noise samples z passing through G to generate the synthetic samples or fake data and then pass through D to minimize $[\log(1 - D(G(z)))]$. The goal of the generator's objective function is to fool D to make a wrong classification as it inspires D to identify the synthetic samples as real ones or to label the synthetic samples as 1. In other words, it attempts to minimize the likelihood that D classifies these samples correctly as fake data. Thus, $[\log(1 - D(G(z)))]$ is reduced when the synthesized samples are wrongly labeled as 1. However, in real practice, G is poor in generating synthetic samples in the early training stage, making D too easy to classify the synthesized samples from G and the real samples from the dataset due to their great difference. To solve this problem, the generator's objective function can be changed from reducing $[\log(1 - D(G(z)))]$ to increasing $[\log D(x)]$ to provide enough gradient for G. This alternative objective function of the generator provides a stronger gradient in the early stage of the generator's training.

As both objective functions are distinct, the two networks are trained together by alternating the gradient updates following the standard gradient rule with the momentum parameter. There are two main procedures for training G and D networks to update their gradients alternately. The process is first to freeze G and train only D. This alternative gradient update is triggered by the fact that the discriminator needs to learn the outputs

of the generator to define the real data from the fake ones. Thus the generator is required to be frozen. The discriminator network can be updated as shown in the following equation:

$$\nabla_{\theta_d} 1/m \sum_{i=1}^{m} \left[\log D\left(x^{(i)}\right) + \log\left(1 - D\left(G\left(z^{(i)}\right)\right)\right)\right] \tag{16.3}$$

Specifically, the gradient updates are different for the learning of two networks: D uses stochastic gradient ascent, while G uses stochastic gradient descent. D uses a hyperparameter k to update steps for each step of G. To update D, the stochastic gradient ascent performs the updates for k times to increase the likelihood that D accurately labels both samples (fake and real data). These updates are achieved using backpropagation on an equal number of examples with batch size of 2. Let noise samples m consist of $\{z^{(1)}, z^{(2)}, \ldots, z^{(m)}\}$ from the distribution of generator $p_g(z)$ and real examples m consist of $\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$ from the distribution of real data p_{data}. Once D is updated, then only G is trained to update its gradient as shown in the following equation:

$$\nabla_{\theta_g} 1/m \sum_{i=1}^{m} \log\left(1 - D\left(G\left(z^{(i)}\right)\right)\right) \tag{16.4}$$

Concerning the update for the generator network, m noise samples are input into G only once to generate m synthesized examples. G uses the stochastic gradient descent to minimize the likelihood that D labels the synthesized samples correctly. The generator's objective function aims to minimize $[\log(1 - D(G(z)))]$ to boost the likelihood that synthetic examples are classified as real examples. This process computes the gradients during backpropagation for both networks. Still, it only updates the parameters of G. D is kept constant during the training of G to prevent the possibility that G might never converge.

16.2.2 The literature of video anomaly detection based on generative adversarial network

Here we review recent literature works that use GANs for anomaly detection in crowds. There are three outstanding video anomaly detection works ranked by publication year as follows.

16.2.2.1 Cross-channel generative adversarial networks

Starting with the work proposed by Ravanbakhsh et al. [20], this work applies conditional GANs (cGANs) where the generator G and the discriminator D are both conditioned to the real data and also relies on the idea of the translating image to image [35]. Following the characteristics of cGANs, the input image x is fed to G to produce a generated image p that looks realistic. G attempts to deceive D that p is real, while D efforts to identify x from p. This paper states that U-Net structure [36] in the generative network

and a patch discriminator (Markovian discriminator) advantage the transformation of images in different representations (e.g., spatial to temporal representations). Thus, the authors adopt this concept to translate the appearance of a frame to the motion of optical flow and target to learn only the normal patterns. To detect an abnormality, they compare the generated image with the real image by using a simple pixel-by-pixel difference along with pretrain on ImageNet [37]. The framework of the anomaly detection in videos using cGANs during testing is shown in Fig. 16.3.

More specifically, the authors train two networks $\mathcal{N}^{F\to O}$ that uses frames F to generate optical flow O and $\mathcal{N}^{O\to F}$ that uses optical flow O to generate frames F. Assume that F_t is the frame of the training video sequence with RGB channels at time t and O_t is the optical flow containing three channels (horizontal, vertical, and magnitude). O_t is obtained from two consecutive frames, F_t and F_{t+1}, following the computation in Ref. [38]. As both generative and discriminative models are the conditional networks, G generates output from its inputs consisting of an image x and noise vector z, providing a synthetic output image $p = G(x,z)$. In the case of $\mathcal{N}^{F\to O}$, x is assigned as a current frame $x = F_t$, hence the generation of its corresponding optical flow (or the synthetic output image p) is represented as $y = O_t$. For D, it takes two inputs, whether it is (x,y) or (x, p) to yield a probability of classes belonging to its pair. The loss functions are defined, including a reconstruction loss $\mathcal{L}_{L1}$ and a conditional GAN loss $\mathcal{L}_{cGAN}$ as shown in Eqs. (16.5) and (16.6), respectively.

For $\mathcal{N}^{F\to O}$, $\mathcal{L}_{L1}$is determined with the training set $\mathcal{X} = \{(F_t, O_t)\}$ as

$$\mathcal{L}_{L1}(x, y) = \|y - G(x, y)\|_1 \tag{16.5}$$

whereas $\mathcal{L}_{\text{cGAN}}$ is assigned as

$$\mathcal{L}_{cGAN}(D, G) = \mathbb{E}_{(x, y)\in X}[\log D(x, y)] + \mathbb{E}_{X\in\{F_t\}, z\in Z}[\log(1 - D(x, G(x, z)))] \tag{16.6}$$

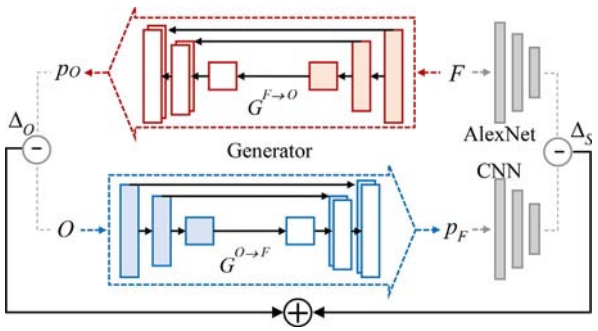

Fig. 16.3 A framework of video anomaly detection using conditional generative adversarial nets (cGANs) during testing in Ref. [20]. There are two generator networks: (i) producing a corresponding optical flow image from its input frames and (ii) reconstructing an appearance from a real optical flow image.

In contrast, the training set of $\mathcal{N}^{O\to F}$ is $\mathcal{X}=\{(O_t, F_t)\}_{t=1}^{N}$. Once the training is finished, the only model being used during testing is G, consisting of $G^{F\to O}$ and $G^{O\to F}$ networks. Both networks are not able to reconstruct the abnormality since they have been trained with only normal events. Then, the abnormality can be found by subtracting pixels to obtain the difference between O and p_o, $\Delta_O = O - p_o$, where p_o is the optical flow reconstruction when using F, $G^{F\to O}(F)$. Another network is $G^{O\to F}(O)$ that produces the appearance reconstruction p_F. However, Δ_O provides more information than the difference between F and p_F. In this case, the authors added an additional network to find the difference in semantic perspective Δ_S by using AlexNet [39] with its fifth convolutional layer h defined as $\Delta_S = h(F) - h(p_F)$. These two difference Δ_O and Δ_S are combined and normalized in between [0,1] for an abnormality map. Finally, the final score of abnormality heatmap is achieved by summing the normalized semantic difference map N_S and the normalized optical flow difference map N_O, $A = N_S + \lambda N_O$ where $\lambda = 2$.

16.2.2.2 Future frame prediction based on generative adversarial network

Apart from the above work, there is an approach for a video future frame prediction of abnormalities based on GAN proposed by Liu et al. [21]. This work is inspired by the problems of anomaly detection that are mostly about minimizing the reconstruction errors from the training data. Instead, the authors proposed an unsupervised feature learning for video prediction and leveraged the distinction of their predicted frame with the real data for anomaly detection. The framework of video future prediction for detecting anomalies is shown in Fig. 16.4. In the training stage, only normal events are learned since they are considered as predictable patterns, using both appearance and motion constraints. Then, during testing, all frames are input and compared with the predicted frame. If the input frame corrects with the predicted frame, it is a normal event. If it is not, then it

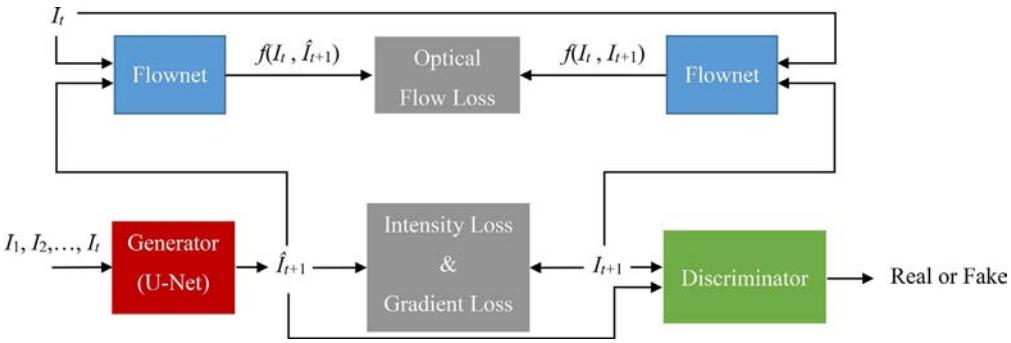

Fig. 16.4 Video future prediction framework for anomaly detection [21]. U-Net structure and pretrained Flownet are used to predict a target frame and to obtain optical flow, respectively. Adversarial training is used to disguise whether a predicted frame is real or fake.

becomes an anomalous event. Using a good predictor to train is the key in this work; thus U-Net network [36] is chosen due to its performance in translating images with the GAN model.

In mathematical terms, given a video sequence containing t frames $I_1, I_2, \ldots, I_t$. In this work, a future frame is defined as I_{t+1}, while a predicted future frame as $\hat{I}_{t+1}$. The goal is to make $\hat{I}_{t+1}$ close to I_{t+1} to determine whether $\hat{I}_{t+1}$ is an abnormal or normal event by minimizing their distance in terms of intensity and gradient. In addition, optical flow is used to represent the temporal features between frames; I_{t+1} and $I_{t,}$ and $\hat{I}_{t+1}$ and $I_{t.}$. We first take a look at the generator objective function L_G, consisting of appearance (ingredient L_{int} and gradient L_{gd}), motion L_{op}, and adversarial training L_{adv}^{G}, in Eq. (16.7):

$$L_G = \lambda_{int} L_{int}\left(\hat{I}_{t+1}, I_{t+1}\right) + \lambda_{gd} L_{gd}\left(\hat{I}_{t+1}, I_{t+1}\right) + \lambda_{op} L_{op}\left(\hat{I}_{t+1}, I_{t+1}, I_t\right) + \lambda_{adv} L_{adv}^{G}\left(\hat{I}_{t+1}\right) \tag{16.7}$$

The discriminator objective function L_D is defined in Eq. (16.8):

$$L_D = L_{adv}^{D}\left(\hat{I}_{t+1}, I_{t+1}\right). \tag{16.8}$$

The authors followed the work in Ref. [40] by using intensity and gradient difference. Specifically, the intensity and gradient penalties assure the similarity of all pixels and the sharpened generating images, respectively. Suppose $\hat{I}_{t+1}$ is $\hat{I}$ and I_{t+1} is I. The distance of ℓ_2 between $\hat{I}$ and I is minimized in intensity to guarantee the similarity in the RGB space as shown in the following equation:

$$L_{int}\left(\hat{I}, I\right) = \left\|\hat{I} - I\right\|_2^2. \tag{16.9}$$

Then the gradient loss is defined as follows [40] in Eq. (16.10):

$$L_{gd}\left(\hat{I}, I\right) = \sum_{i,j} \left\| \left|\hat{I}_{i,j} - \hat{I}_{i-1,j}\right| - \left|I_{i,j} - I_{i-1,j}\right| \right\|_1 + \left\| \left|\hat{I}_{i,j} - \hat{I}_{i,j-1}\right| - \left|I_{i,j} - I_{i,j-1}\right| \right\|_1 \tag{16.10}$$

where i and j are the frame index.

Then the optical flow estimation is applied by using a pretrained network, Flownet [41], denoted as f. The temporal loss is defined in the following equation:

$$L_{op}\left(\hat{I}_{t+1}, I_{t+1}, I_t\right) = \left\| f\left(\hat{I}_{t+1}, I_1\right) - f\left(I_{t+1}, I_1\right) \right\|_1. \tag{16.11}$$

In the manner of the adversarial network, the training is an alternative update. The U-Net is used as the generator, while a patch discriminator is used for the discriminator following Ref. [35]. To train the discriminator D, they assign a label of 0 to a fake image and a label of 1 to a real image. The goal of D is to categorize the real future frame I_{t+1} into class 1 and the predicted future frame $\hat{I}_{t+1}$ into class 0. During training the discriminator

D, the weight of G is fixed by using the loss function of mean square error (MSE), denoted as $L_{MSE.}$ Hence, the L_{MSE} of D can be defined in the following equation:

$$L_{adv}^{D}(\hat{I}, I) = \sum_{i,j} L_{MSE}\left(D(I_{i,j}),\, 1\right)/2 + \sum_{i,j} L_{MSE}\left(D(\hat{I}_{i,j}), 0\right)/2, \tag{16.12}$$

where i and j are the patch index. The MSE loss function L_{MSE} is defined in the following equation:

$$L_{MSE}(\hat{Y}, Y) = \left(\hat{Y} - Y\right)^2, \tag{16.13}$$

where the value of Y is in [0,1], while $\hat{Y} \in [0, 1]$.

In contrast, the objective of the generator G is to reconstruct images to fool D to label them as 1. The weight of D is fixed during training G. Thus, L_{MSE} of G is defined as shown in the following equation:

$$L_{adv}^{G}(\hat{I}) = \sum_{i,j} L_{MSE}\left(D(\hat{I}_{i,j}),\, 1\right)/2. \tag{16.14}$$

To conclude, the appearance, motion, and adversarial training can assure that normal events are generated. The events with a great difference between the prediction and the real data are classified as abnormalities.

16.2.2.3 Cross-channel adversarial discriminators

As in Ref. [20], the authors have been continuously proposed another GAN-based approach [22] for abnormality detection in crowd behavior. The training procedure is the same as Ref. [20] that only the frames of normal events are trained with the cross-channel networks based on conditional GANs by engaging G to translate the raw pixel image to the optical flow, inspired by Isola et al. [35]. This paper takes advantage of the U-Net framework [36] for translating from one to another image and representing a multichannel data, i.e., spatial and temporal representations, into an account, similarly to Refs. [20] and [21]. The novel part is in the testing where the authors proposed the end-to-end framework without additional classifiers by using the learned discriminator as the classifier of abnormalities. The framework of the cross-channel adversarial discriminators is shown in Fig. 16.5. For a brief explanation, G and D are simultaneously trained only on the frames of normalities. G generates the synthetic image from the learned normal events, while D learns how to differentiate whether its input is normal events or not due to the data distribution, defining the abnormal events as outliers in this sense. During testing, only D is being used directly to classify anomalies in the scene. In such a way, there is no need for reconstructing images at the testing time unlike the common GAN-based models [20, 21] that use G in testing.

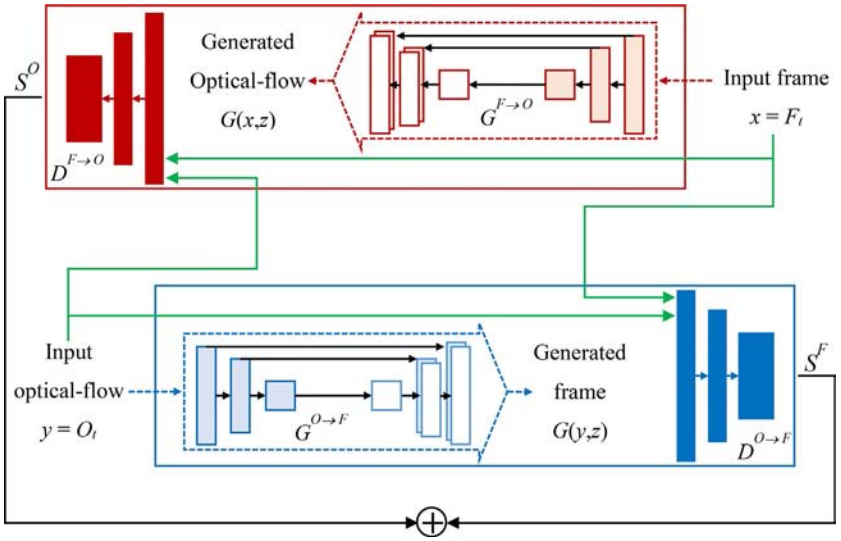

Fig. 16.5 Cross-channel adversarial discriminators flow diagram with additional detail on parameters following [22]. Two generator networks are used during training: (i) generating a corresponding optical flow image and (ii) reconstructing an appearance. At testing time, only discriminative networks are used and represented as a learned decision boundary to detect anomalies.

Specifically, there are two networks used for training: $\mathcal{N}^{F\to O}$ and $\mathcal{N}^{O\to F}$. Suppose that F_t is a frame (at time t) of a video sequence and O_t is an optical flow acquired from two consecutive frames, F_t and F_{t+1}, following the computation of optical flow-based theory for warping [38]. In this work, G and D are conditioned to each other. G takes an image x and noise vector z to output a synthetic optical flow $r = G(x,z)$. For $\mathcal{N}^{F\to O}$, let x be a current frame $x = F_t$. Then, the generation of its corresponding optical flow r can be represented as $y = O_t$. Conversely, D takes two inputs, whether it is (x,y) or (x,r) to obtain a probability of classes belonging to its pair. The reconstruction loss $\mathcal{L}_{L1}$ and a conditional GAN loss $\mathcal{L}_{cGAN}$ can be obtained as follows.

In the case of $\mathcal{N}^{F\to O}$, $\mathcal{L}_{L1}$ is determined with the training set $\mathcal{X} = \{(F_t, O_t)\}$ as shown in the following equation:

$$\mathcal{L}_{L1}(x, y) = \|y - G(x, y)\|_1 \tag{16.15}$$

whereas $\mathcal{L}_{\text{cGAN}}$ is represented in the following equation:

$$\mathcal{L}_{\text{cGAN}}(D, G) = \mathbb{E}_{(x,y)\in X}[\log D(x, y)] + \mathbb{E}_{X\in\{F_t\}, z\in Z}[\log(1 - D(x, G(x, z)))]. \tag{16.16}$$

In contrast, the training set of $\mathcal{N}^{O\to F}$ is $\mathcal{X} = \{(O_t, F_t)\}_{t=1}^{N}$. Note that all training procedures are the same as Ref. [20]. G performs as implicit supervision for D. Both $G^{F\to O}$ and $G^{O\to F}$ networks lack the ability to reconstruct the abnormal events because they observe only the normal events during training, while $D^{F\to O}$ and $D^{O\to F}$ have learned the patterns to distinguish real data from artifacts.

The discriminator is considered as the learned decision boundary that splits the densest area (i.e., the normal events x_3) from the rest (i.e., abnormal events x_1 and generated images x_2). Since the goal is to detect abnormal events x_1, the latter outside the decision boundaries are judged as outliers by D. During testing, the authors focus on only the discriminative networks for two-channel transformation tasks. The patch-based discriminators $\hat{D}^{F\to O}$ and $\hat{D}^{O\to F}$ are applied to the test frame F and its corresponding optical flow O with the same 30×30 grid, resulting in two 30×30 score maps represented as S^O for $\hat{D}^{F\to O}$ and S^F for $\hat{D}^{O\to F}$. In detail, a patch p_F on F and a patch p_O on O are input to $\hat{D}^{F\to O}$. Any abnormal events occur in these patches (p_F and/or p_O) are considered outliers according to the distribution learned by $\hat{D}^{F\to O}$, resulting in a low probability score of $\hat{D}^{F\to O}(pF, pO)$. To finalize the anomaly maps, the normalized channel score maps with equal weights are fused $S = S^O + S^F$ in the range [0,1] and then applied with a range of thresholds to compute the ROC curves.

We notice that there are some interesting points in this work: (i) the authors state that $D^{F\to O}$provides higher performance than $D^{O\to F}$ since the input of $D^{F\to O}$ is the real frame which contains more information than the optical flow frame, (ii) their proposed end-to-end framework is simpler and also faster in testing than Ref. [20] since they do not require the generative models during testing and any additional classifiers to add on top of the model such as a pretrained AlexNet [39]. The observations from the early works inspire us to propose our method [26], which we discuss in Section 16.3.2.

16.3 Training a generative adversarial network

16.3.1 Using generative adversarial network based on the image-to-image translation

The anomalous object observation using an unsupervised learning approach is considered as the structural problem of the reconstruction model known as the per-pixel classification or regression problem. The common framework used to solve this type of problem and explored by the state-of-the-art works in anomaly detection task [20–22] is the generative image-to-image translation network constructed by Isola et al. [35]. In general, they use this network to learn an optimal mapping from input to output image based on the objective function of GAN. Their experimental results prove that the network is good at generating synthesized images such as color, object reconstruction from edges, and label maps.

From an overall perspective, based on the original GANs [27], the input of G consists of the image x and noise vector z in which the mapping of the original GANs is represented as G: $z\to y$ (learning from z to output image y). Differently, conditional GANs learn from two inputs, x and z, to y, represented as G: $\{x,z\}\to y$. However, z is not necessary for the network as G still can learn the mapping without z, especially in the early training where G is learned to ignore z. Thus, the authors decided to use z in

the form of dropout in both the training and testing process. Considering the objective function, since the architecture of image-to-image translation network is based on the conditional GANs, its objective functions are indicated the same as we explained in Refs. [20] (see Eqs. 16.5 and 16.6) and [22] (see Eqs. 16.15 and 16.16) where two inputs are required for the discriminator $D(x,y)$ and L1 loss is used to help output sharper image. On the other hand, the future frame prediction [21] applied unconditional GANs that uses only one input for the discriminator $D(y)$ (see Eq. 16.12). It relied on the traditional L2 regression (MSE loss function) to condition the output and the input. This forcing condition results in lower performance (i.e., blurry images) on the frame-level anomaly detection compared to Refs. [20] and [22].

16.3.2 Unsupervised learning of generative adversarial network for video anomaly detection

In this section, we introduce our proposed method, named DSTN [26]. We take advantage of image-to-image transformation architecture with the U-Net network [36] to translate a spatial domain to a temporal domain. In this way, we can obtain comprehensive information on the objects from both appearance and motion information (optical flow). The proposed DSTN differs from the previous works [20–22] since we focus on only one deep spatiotemporal translation network to enhance the anomaly detection performance at a frame level and the challenging anomaly localization at a pixel level with regard to accuracy and computational time. Specifically, we include preprocessing and postprocessing stages to assist the learning of GAN without using any pretrained network to help in the classification, making the DSTN faster and more flexible. Besides, we differ from Ref. [35] as our target output is the motion information of the object corresponding to its appearance, not the realistic images. There are two main procedures for each training and testing time. For training, a feature collection and a spatiotemporal translation play essential roles to sufficiently collect information and effectively learn to model, respectively. Then a differentiation and an edge wrapping are utilized at the testing time. We shall explain the main components of our proposed method in detail including the system overview for both training and testing as follows.

16.3.2.1 System overview

We first start with the system overview of ourDSTN. The DSTN is based on GAN and plugged with preprocessing and postprocessing procedures to improve the performance in learning normalities and localizing anomalies. Overall, the main components of the DSTN are fourfold: a feature collection, a spatiotemporal translation, a differentiation, and an edge wrapping. The feature collection is a key initial process for extracting the appearances of objects. These features are fed into the model to learn the normal patterns. In our case, the generator G is used in both training and testing time, while the discriminator D only at the training time. During training, G learns the normal patterns from the

training videos. Hence it understands and has only the knowledge of what normal patterns look like. The reason why we feed only the frames of normal patterns is that we need the model to be flexible and able to handle all possible anomalous events in real-world environments without labels of anomalies. In testing, all videos, including normal and abnormal events, are input into the model where G tries to reconstruct the appearance and the motion representation from the learned normal events. Since G has not learned any abnormal samples, it is unable to reconstruct the abnormal area properly. We then take this inability to correctly reconstruct anomalous events to detect the anomalies in the scene. The anomalies can be exposed by subtracting pixels in the local area between the synthesized image and the real image and then applying the edge wrapping at the final stage to achieve precise edges of the abnormal objects.

Specifically, during training, only normal events of original frames f are input with background removal frames f_{BR} into the generative network G, which contains encoder En and decoder De, to generate dense optical flow (DIS) frames OF_{gen} representing the motion of the normal objects. To attain good optical flow, the real DIS optical flow frame OF_{dis} and f_{BR} are fused to eliminate noise that frequently occurs in OF_{dis}, giving Fused Dense Optical Flow frames OF_{fus}. The patches of f and f_{BR} are concatenated and fed into G to produce the patches of OF_{gen}, while D has two alternate inputs, the patches of OF_{fus} (real optical flow image) and OF_{gen} (synthetic optical flow image), and tries to discriminate whether OF_{gen} is fake or real. The training framework of DSTN is shown in Fig. 16.6.

After training, the DSTN model understands a mapping of the appearance representation of normal events to its corresponding dense optical flow (motion representation). All parameters used in training are also used in testing. During testing, the unknown events from the testing videos will be reconstructed by G. However, the reconstruction of G provides results of unstructured blobs based on its knowledge of the learned normal patterns. Thus, these unstructured blobs are considered as anomalies. To capture the anomalies, the differentiation is computed by subtracting the patches of OF_{gen} with the patches of OF_{fus}. Note that not only the anomaly detection is significantly essential for real-world use, but also the anomaly localization. Therefore, edge wrapping (EW) is proposed to obtain the final output by retaining only the actual edges corresponding to the real abnormal objects and suppressing the rest. The DSTN framework at the testing time is shown in Fig. 16.7.

16.3.2.2 Feature collection

We explain our proposed DSTN based on its training and testing time. During training, even GAN is good at data augmentation and image generation on small datasets, it still desires for sufficient features (e.g., appearance and motion features of objects) from data examples to feed its data-hungry characteristics of the deep learning-based model. The importance of feature extraction is recognized and represented as the preprocessing

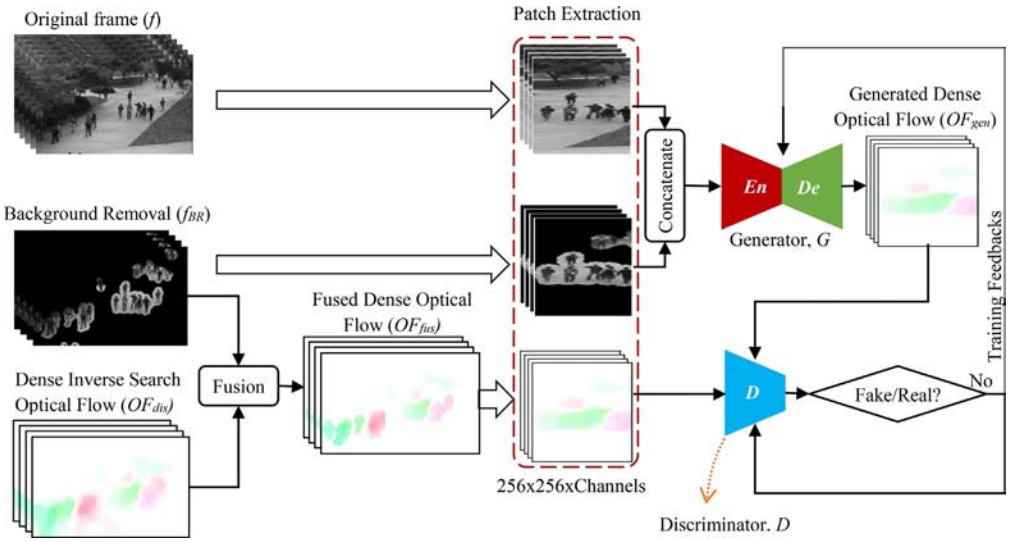

Fig. 16.6 A training framework of proposed DSTN.

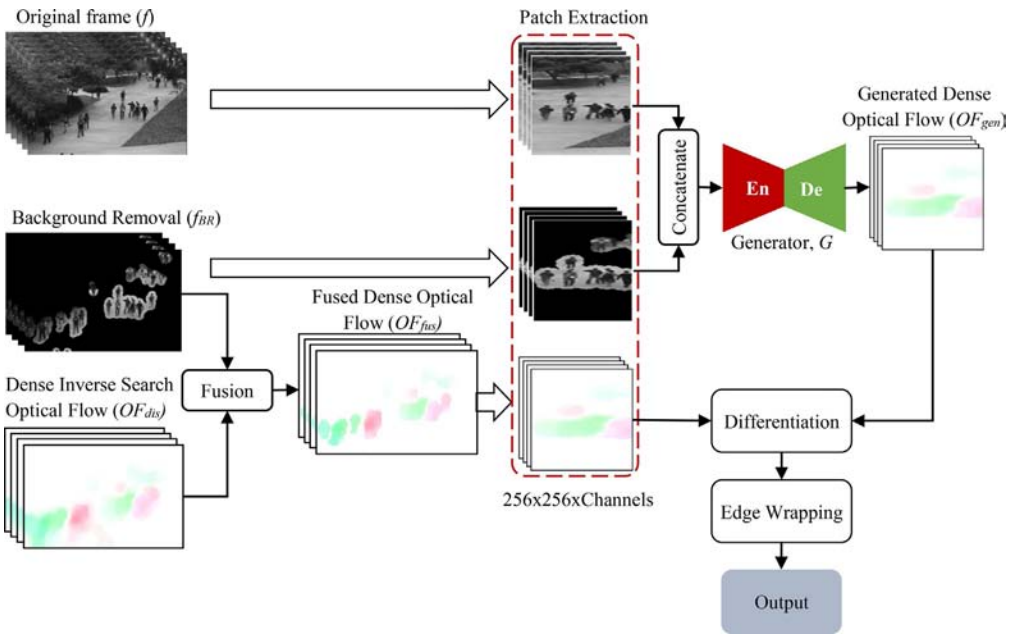

Fig. 16.7 A testing framework of proposed DSTN [26].

procedure before learning the model. There are several procedures in the feature collection, including (i) background removal, (ii) fusion, (iii) patch extraction, and (iv) concatenated spatiotemporal features, as described below.

In (i) background removal, we only take the moving foreground objects into account because we focus on the real situation from the CCTV cameras. Thus the static background is ignored in this sense. This method benefits in extracting the object features and removing irrelevant pixels in the background for obtaining only the important appearance information. Let f_t be the current frame at time t of the video and f_{t-1} be the previous frame. The background removal f_{BR} is computed by using the frame absolute difference as shown in the following equation:

$$f_{BR} = |f_t - f_{t-1}| \tag{16.17}$$

After computing the frame absolute difference, the background removal output is binarized and then concatenated with the original frame f to acquire more information on the appearance, assisting the learning of the generator. It can simply conclude that more input features mean the better performance of the generator. The significance of the concatenated f_{BR} and f frames is that it delivers extra features on the appearance of foreground objects in f_{BR} as f contains all information where f_{BR} may lose some of the information during the subtraction process.

For (ii) fusion, according to the literature on GAN for video anomaly detection [20–22], they all apply theory for warping [38] for representing the temporal features. However, it has problems in obtaining all information on objects and also with high time complexity. Since the development of video anomaly detection requires reliable performance in terms of accuracy and running time, thus the theory for warping is not suitable for this task. To achieve the best performance on motion representation, we use dense inverse search (*DIS*) [42] to represent the motion features of foreground objects in surveillance videos due to its high accuracy and low time complexity performance on detecting and tracking objects. The *DIS* optical flow OF_{dis} can be obtained from two consecutive frames f_t and f_{t-1} as shown in Fig. 16.8 where the resolution of f_t and f_{t-1} frames is $238 \times 158 \times$ channels following the UCSD Ped1 dataset [4]. The number of the channels c_p for f_t and f_{t-1} frames is 1 ($c_p = 1$), while c_p of OF_{dis} is 3 ($c_p = 3$).

However, OF_{dis} contains noise dispersed in the background same as the objects as shown in Fig. 16.9. Thus, we propose a novel fusion between OF_{dis} and f_{BR} to use the good foreground object from f_{BR} with OF_{dis} for acquiring both appearance and motion information and assisting in noise reduction in OF_{dis}. The fusion provides a clean background and explicit foreground objects. From Fig. 16.9, it is clearly shown that fusion effectively helps to remove noise from OF_{dis}. Specifically, the noise reduction is implemented efficiently by observing f_{BR} values equal to 0 or 255 and then using

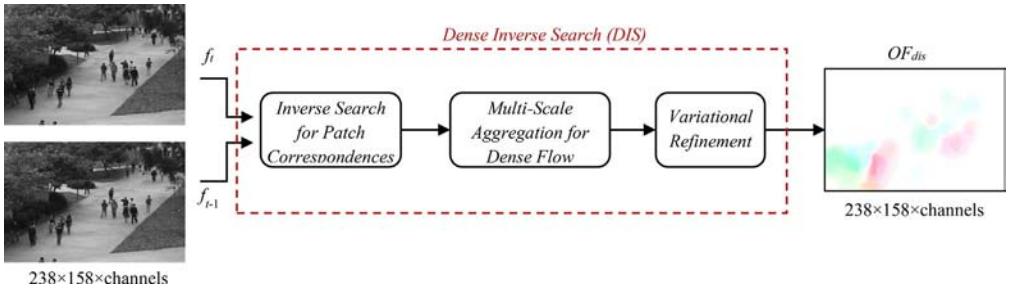

Fig. 16.8 Dense inverse search optical flow framework.

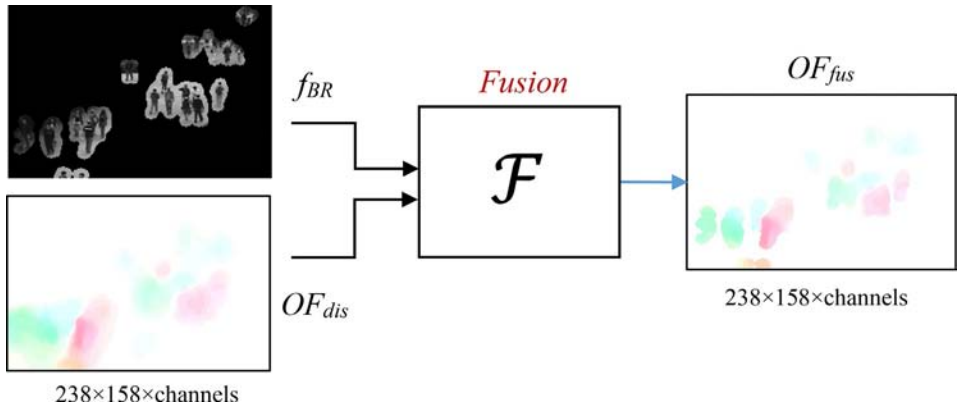

Fig. 16.9 A fusion between background subtraction and real optical flow.

masking of f_{BR} on OF_{dis} to change its values. Let ζ be a constant value. The new output represented OF_{dis} is a fusion OF_{fus} as defined in Eq. (16.18):

$$OF_{fus} = OF_{dis} \lfloor f_{BR} / (f_{BR} + \zeta) \rfloor. \tag{16.18}$$

Apart from (i) background removal and (ii) fusion, (iii) patch extraction acts a part in the feature collection process and supports in acquiring more spatial and temporal features of the moving foreground object in local pixels. By doing that, it can achieve better information than directly extracting features from the full image. Patch extraction is implemented using the full-size of the moving foreground object appearance at the current frame f along with its direction, motion, and magnitude from the frame-by-frame dense optical flow image. We normalize all patch elements in the range $[-1, 1]$. The patch size is defined as $(w/a) \times h \times c_p$, where w is the frame width, h is the frame height, a is a scale value, and c_p is the number of channels. A sliding-window method with a stride d is applied on the input frames of the generator G (i.e., f and f_{BR}) and the discriminator D (i.e., OF_{fus}). Fig. 16.10 shows examples of patch extraction. We extract the patches

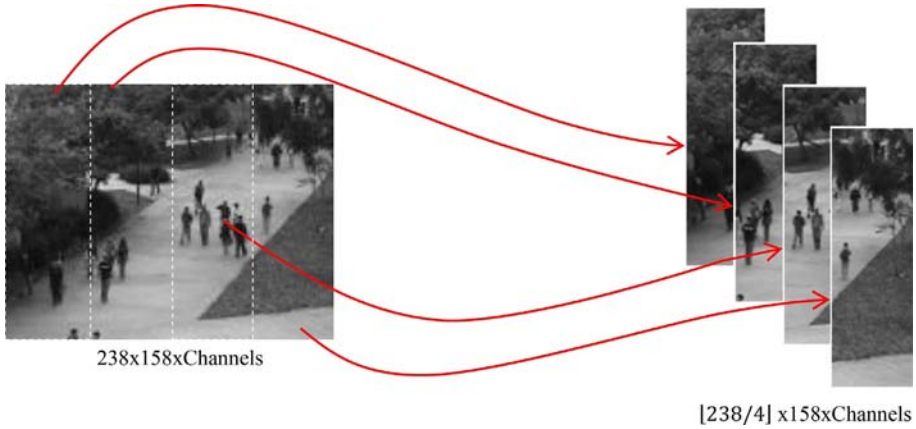

Fig. 16.10 Examples of patch extraction on a spatial frame.

with the scale value $a = 4$ and $d = w/a$ to obtain more local information from spatial and temporal representations. Then, the extracted patch image is scaled up to 256×256 full image to gain more information on the appearance from the semantic information and input into the model for the further process.

The final process of the feature collection is (iv) the concatenation of spatiotemporal features for data preparation. We input the appearance information to the generative model to output the motion information in both training and testing time, as shown in Fig. 16.11. Since providing sufficient feature inputs to G is significantly important to produce good corresponding optical flow images; thus, the patches of f and f_{BR} are concatenated to cover all possible low-level appearance information of the normal patterns for G to understand and learn them extensively. More specifically, f_{BR} provides precise foreground object contours, whereas f provides inclusive knowledge for the whole scene. The sizes of the input and target images are fixed to the 256×256 full image as the default value in our proposed framework. The concatenated frames have two channels

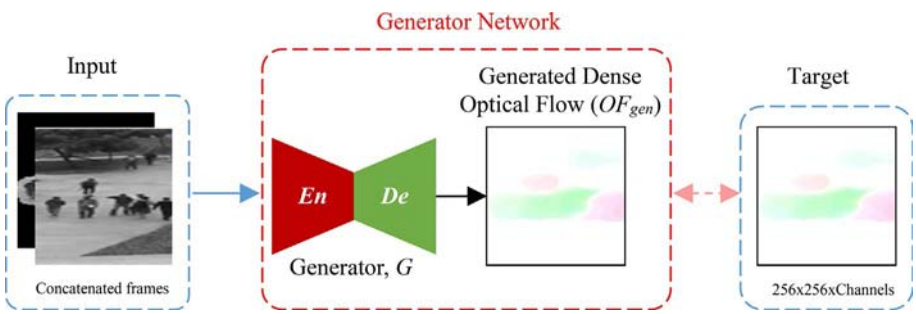

Fig. 16.11 Overview of our data preparation showing spatiotemporal input features (concatenated patches) and output feature (generated dense optical flow patch).

($c_p = 2$) and the temporal target output has three channels ($c_p = 3$). As a final point, this concatenation process implies the potential of the learning of a spatiotemporal translation model to reach its desired temporal target output.

16.3.2.3 Spatiotemporal translation model

In this section, we declare our training structure in the manner of GAN-based U-Net architecture [36] for translating the spatial inputs (f and f_{BR}) to the temporal output (OF_{gen}) and present on how the interplay between the generative and discriminative networks works during training. The details of the proposed spatiotemporal translation model are clearly explained as the following.

The generative network G performs an image transformation from the concatenated f and f_{BR} appearances to the OF_{gen} motion representations. Generally, there are two inputs for G, an image x and noise z, to generate image e as the output with the same size of x but different channels, $e = G(x,z)$ [27, 43, 44]. On the other hand, the additional Gaussian noise z is not prominent to G in our case since G can learn to ignore z in the early stage of training. In addition, z is not that effective for transforming the spatial representation data from its input to the temporal representation data. Therefore, dropout [35] is applied in the decoder within batch normalization [45] instead of z, resulting in $e = G(x)$.

On closer inspection, the full network of generator consisting of the encoder *En* and decoder *De* architectures is constructed by skip connections or residual connections [35], as shown in Fig. 16.12. The idea of skip connections is that it links the layers from the encoder straight to the decoder, allowing the networks to be easier for the optimization

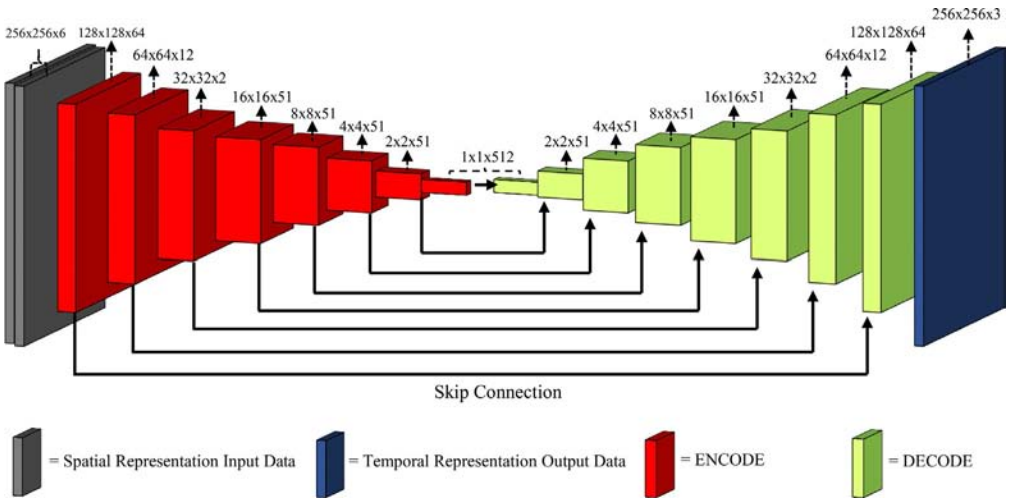

Fig. 16.12 Generator architecture consisting of an encoder and a decoder with skip connections [26].

and providing greater quality and less complexity for image translation than the traditional CNN architectures, e.g., AlexNet [39] and VGG nets [46, 47]. More specifically, let t be the total number of layers in the generative network. The skip connections are introduced at each layer i of En and layer $t - i$ of De. The data can be transferred from the first to the final layer by integrating channels of i with $t - i$. The architectures of En and De are illustrated in Fig. 16.13. En compresses the spatial representation of data to higher-level representation, while De performs the reverse process to generate OF_{gen}. En uses Leaky-ReLU (L-ReLU) activation function. Conversely, De uses the ReLU activation function that benefits in accelerating the learning rate of the model to saturate color distributions [36]. To achieve accurate OF_{gen}, the objective function is assigned and optimized by the Adam optimization algorithm [48] during training.

For the encoder module, it acts as a data compression from a high-dimensional space into a low-dimensional latent space representation to pass to the decoder module. The first layer in the encoder is convolution using CNNs as the learnable feature extraction instead of the handcrafted features approach that is more delicate to derive the obscure data structures than the deep learning approach. The convolution is a linear operation implemented by covering an $n \times n$input image I with a $k \times k$sliding window w. The output of convolution function on cell c of the image I is defined as shown in the following equation:

$$y_c = \sum_{i=1}^{k \times k} (w_i \times I_{i,c}) + b_c \tag{16.19}$$

where y_c is the output after the convolution and b_c is the bias. Let p be the padding and s be the stride. The output size of convolution O is calculated as shown in the following equation:

$$O = (n - k + 2p)/s + 1 \tag{16.20}$$

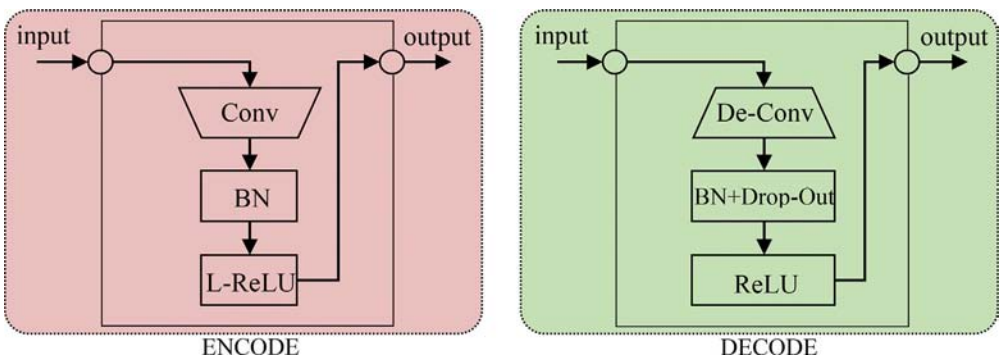

Fig. 16.13 Encoder and decoder architectures [26].

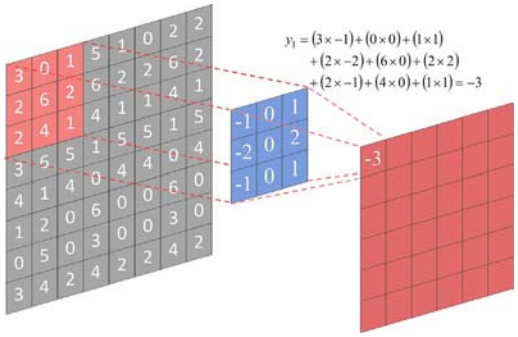

Fig. 16.14 An example of a convolution operation on an image cell.

The convolution operation on an image cell with $b=0$, $n=8$, $k=3$, $p=0$, and $s=1$ is delivered in Fig. 16.14 for more understanding.

Once the convolution operation is completed, batch normalization is applied by normalizing the convolution output following the normal distribution in Ref. [45] to reduce the training time and avoid the vanishing gradient problem. Suppose y is convolution output values over a mini-batch: $B=\{y_1, y_2, \ldots, y_m\}$, γ and β are learnable parameters and$\varepsilon$is a constant to avoid zero variance. The normalized output S can be obtained by scaling and shifting as defined in the following equation:

$$S_i = \gamma \times \hat{y}_i + \beta \equiv BN_{\gamma,\beta}(y_i) \tag{16.21}$$

- Normalize: $\hat{y}_i = (y_i - \mu_B)/\sqrt{\sigma_B^2 + \varepsilon}$,
- Mini-batch mean: $\mu_B = 1/m\sum_{i=1}^{m} y_i$,
- Mini-batch variance: $\sigma_B^2 = 1/m\sum_{i=1}^{m}(y_i - \mu_B)^2$.

The final layer of *En* is adopted by the activation function to introduce nonlinear mapping function from the input to the output (response variable). The nonlinear mapping performs the transformation from one to another scale and decides whether the neurons can pass through it. The nonlinearity property makes the network more complex, resulting in a stronger model for learning the complex input data. The L-ReLu is assigned in the proposed DSTN to avoid the vanishing gradient problem. The property of L-ReLu is to allow the negative value to pass through the neuron by mapping it to the small negative value of the response variable, leading to the improvement of the flow of gradients through the model. The L-ReLu function can be determined, as shown in Eq. (16.22):

$$f(s) = \begin{cases} s, & \text{if } s \geq 0 \\ as, & \text{otherwise} \end{cases} \tag{16.22}$$

where s is the input, a is a coefficient ($a=0.2$) allowing the negative value to pass through the neuron, and $f(s)$ is the response variable. Fig. 16.15 shows the graph representing the input data s mapping to the response variable.

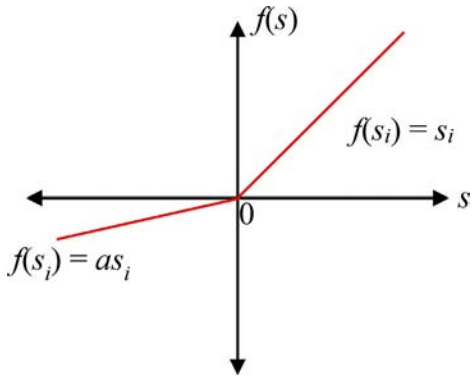

Fig. 16.15 Leaky-ReLu activation function.

Regarding the decoder module *De*, it is an inverse process of encoding part in which the residual connections are passed through the encoder to the corresponding layers at the decoder. The dropout is used in *De* to represent noise vector z as it eliminates the neuron connections using a default probability, helping to prevent the overfitting during training and improve the performance of GAN. Let $h \in \{1, 2, \ldots, H\}$ be the hidden layers of the network, $z^{(h+1)}$be the output layers $h+1$, and $r_j^{(h)}$ be the random variable following Bernoulli distribution with probability p [49]. The feed-forward operation can be described in the following equation:

$$\begin{aligned} r^{(h)} &: Bernoulli(p), \\ \widetilde{f}(s)^{(h)} &= r^{(h)} * f(s)^{(h)}, \\ z_i^{(h+1)} &= w_i^{(h+1)} \widetilde{f}(s)^{(h)} + b_i^{(h+1)}. \end{aligned} \tag{16.23}$$

Apart from the generator, we shall discuss the discriminative network D for the testing procedure. D distinguishes the real patch OF_{fus} ($\gamma = OF_{fus}$) and the synthetic patch OF_{gen} ($OF_{gen} = e$). As a result, D delivers a scalar output determining the possibility that its inputs are from the real data. In the discriminative architecture, PatchGAN is constructed and applied to each partial image to help accelerate the training time for GAN, resulting in a better performance than using the full image discriminator net with the resolution of 256×256 pixels. The discriminator D is implemented by subsampling from 256×256 OF_{fus} image to 64×64 pixels, providing 16 patches of OF_{fus} passing through the Patch-GAN model to classify whether OF_{gen} is real or fake as shown in Fig. 16.16. The reason why we use the 64×64 PatchGAN is that it provides good pixel accuracy and good intensity on the appearance, making the synthetic image to be more recognizable. The experimental results on the impact of using the 64×64 PatchGAN can be found in Ref. [26].

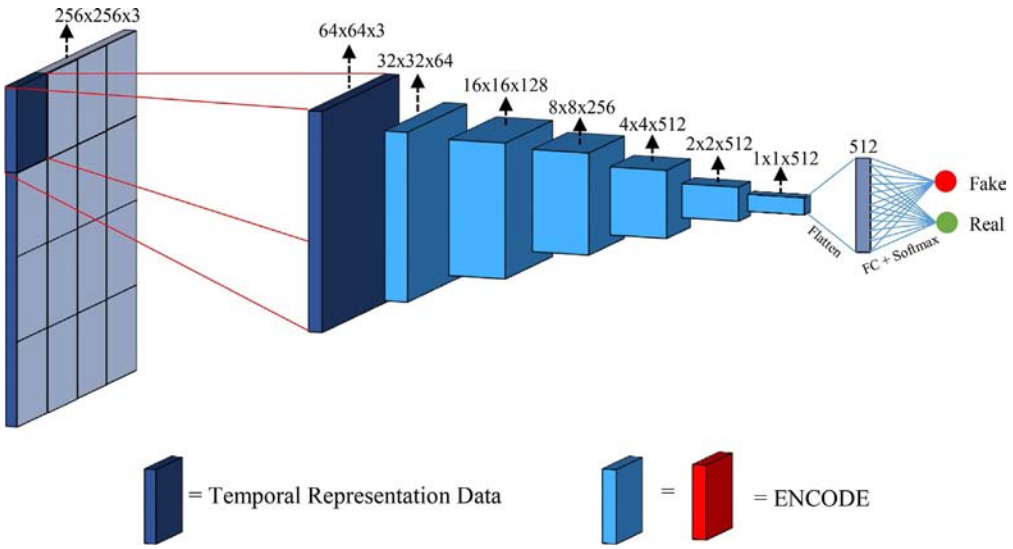

Fig. 16.16 Discriminator architecture with PatchGAN model [26].

To define objective function and optimization, we first discuss two objective functions determined during training; a GAN Loss, L_{GAN}, and a L1 Loss or a generator loss, L_{L1}. Note that our proposed DSTN method comprises only one translational network from the spatial (appearance) to the temporal (motion) image representations. The motion representation is computed based on the dense optical flow using arrays of horizontal and vertical components with the magnitude. Let y be the output image OF_{fus}, x be the input image for G (the concatenated f and f_{BR} image), and z is the additional Gaussian noise vector. Specifically, the dropout is adopted as z, then G can be represented as $G(x)$. The objective functions can be denoted in Eqs. (16.24) and (16.25):

- GAN Loss:

$$L_{GAN}(G, D) = E_y[\log D(y)] + E_x[\log(1 - D(G(x)))] \tag{16.24}$$

- L1 Loss:

$$L_{L1}(G) = E_{x,y}\left[\|y - G(x)\|_1\right] \tag{16.25}$$

Then, the optimization of G can be defined as in the following equation:

$$G^* = \arg\min_G \max_D L_{GAN}(G, D) + \lambda L_{L1}(G) \tag{16.26}$$

The advantage of using one spatiotemporal translation network is that it has less complexity while providing sufficient important features of objects for the learning of GAN.

16.3.3 Anomaly detection

After the training, the spatiotemporal translation network perceives the transformation from the concatenated f and f_{BR} appearance to the OF_{fus} motion representations. All parameters in training are applied in the testing. To detect anomalies, we input two consecutive frames (f and f_{t-1}) from the test videos to the model. During testing, G is used to reconstruct OF_{gen} following its trained knowledge. However, since G has trained with only the normal patterns, G is unable to regenerate the unknown events the same as the normal ones. We take this generator's inability to reconstruct correct abnormal events in order to detect all possible anomalies that occur in the scene. The anomalies can be exposed by subtracting the patches of OF_{fus} and OF_{gen} to locate the difference in local pixels. To be more accurate on the object localization, edge wrapping is proposed to highlight the actual local pixels of anomalies.

For anomaly detection, differentiation is a simple and effective method to obtain abnormalities. The pixels between a patch of OF_{fus} (real image) and a patch of OF_{gen} (fake image) are subtracted to determine even if there are anomalous events in the scene. This differentiation is directly defined in in the following equation:

$$\Delta_{OF} = OF_{fus} - OF_{gen} > 0 \tag{16.27}$$

where Δ_{OF} is the differentiation output in which its value is greater than 0 ($\Delta_{OF} > 0$). The reason why Δ_{OF} can successfully indicate the abnormal events in the scene is that the differentiation between OF_{fus} and OF_{gen} provides a large difference in the anomalous areas where G is unable to reconstruct the abnormal events in OF_{gen} the same as the abnormal events in OF_{fus} (the real abnormal object from testing video sequence). In other words, G tries to reconstruct OF_{gen} the same as OF_{fus}, but it can only reconstruct unstructured blobs based on its knowledge of the learned normal events, making the abnormal events of OF_{gen} different from OF_{fus}. Δ_{OF} provides the score indicating the probability of pixel, whether it belongs to normalities or abnormalities. The range of pixel values for each Δ_{OF} from the test videos is between 0 and 1 where the highest pixel value is considered as an anomaly. To normalize the probability score from Δ_{OF}, the maximum value M_{OF} of all components is computed following the range of pixel values for each test video. From this process, we can gradually alter the threshold of the probability scores of anomalies to define the best decision boundary for obtaining ROC curves. Suppose the position of the pixel in the image is (i,j). The normalization of Δ_{OF} denoted as N_{OF} is shown in the following equation:

$$N_{OF}(i,j) = 1/M_{OF}\Delta_{OF}(i,j) \tag{16.28}$$

However, even we obtain good normalized differentiation N_{OF} showing anomalies in the scene, there are concerning problems occurred in the experimental results, such as a false positive detection on the normal events (i.e., normal event is detected as abnormal

events) and overdetection on the pixels around the actual abnormal object (i.e., the area of the detected abnormal object is too large). This is because the performance of object localization is not effective enough. Therefore, we propose the edge wrapping (*EW)* method to overcome these concerning problems and specifically enhance the pixel-level anomaly localization performance. Our *EW* is performed by using [50] to preserve only the edges of the actual abnormal object and suppressing the rest (e.g., noise and insignificant edges that do not belong to the abnormal object), providing precise abnormal event detection and localization. *EW* performs as a multistage process categorized into three phases: a noise reduction, an intensity gradient, and a nonmaximum suppression.

To eliminating background noise and irrelevant pixels of abnormal objects, a Gaussian filter is applied to blur the normalization of differentiation N_{OF} with the size of $w_e \times h_e \times c_e$ where w_e and h_e are the width and the height of the filter and c_e is the number of channels (e.g., the gray scale image has $c_e = 1$ and the color image has $c_e = 3$). Our differentiation output is the gray scale image $c_e = 1$. Considering the intensity gradient, an edge gradient G_e is achieved using a gradient operator to filter the image in a horizontal direction G_x and a vertical direction (G_y) for obtaining gradient magnitude perpendicular to edge direction at each pixel. The derivative filter has the same size as the Gaussian filter. The first derivative is computed, as shown in Eqs. (16.29) and (16.30):

$$G_e = \sqrt{G_x^2 + G_y^2} \tag{16.29}$$

$$\theta = \tan^{-1}\left(G_y / G_x\right) \tag{16.30}$$

Then, a threshold is defined to conserve only the significant edges. This process is known as nonmaximum suppression. In this phase, the gradient magnitude at each pixel is investigated whether it is greater than a threshold T where we use a value of 50 as it performs the best results, as discussed in Ref. [26]. If it is greater than T, it shows an edge point corresponding to a local maxima to all possible neighborhoods. Hence, we preserve the local maxima and suppress the rest to 0 to acquire the edges corresponding to the certain anomalies. In addition, the Gaussian filter is then again applied with a kernel size of $w_e \times h_e \times c_e$ to prevent noise in the image, representing an output *EW* for the final output of the anomaly localization *OL* where ζ is a constant value. The anomaly localization *OL* can be computed, as shown in the following equation:

$$OL = \Delta_{OF} \lfloor EW / EW + \zeta \rfloor \tag{16.31}$$

16.4 Experimental results

The performance of the DSTN is evaluated on the publicly standard benchmarks used in the video anomaly detection task, UCSD pedestrian [4], UMN [1], and CUHK Avenue [6]. These datasets are recorded in crowds containing indoor and outdoor scenes. Our

experiment results are compared with various competing methods respecting the accuracy on both frame and pixel level and the computational time. Additionally, we indicate the impact of GAN based U-Net network with residual connections compared with another popular architecture, i.e., autoencoder, and address the advantages and the disadvantages of GAN for anomaly detection. The detail of each subtopic is explained as follows.

16.4.1 Dataset

16.4.1.1 UCSD dataset

The UCSD pedestrian dataset [4] consists of walking crowded pedestrian in two outdoor scenes with various anomalies, e.g., cycling, skateboarding, driving vehicles, and rolling wheelchairs. It is a well-known video benchmark dataset for the anomaly detection task due to its complex scene in the real environments with the low-resolution images. There are two subfolders: Ped1 and Ped2, where Ped stands for the pedestrian. The UCSD Ped1 contains 5500 normal frames with the 34 training video sequences and 3400 anomalous frames with 16 testing video sequences. The image resolution of the UCSD Ped1 is 238×158 pixels, and the UCSD Ped2 is 360×240 pixels for all frames. In the UCSD Ped2, there are 346 frames for normal events with 16 training video sequences and 1652 frames for anomalous events with 12 testing video sequences. The characteristics of Ped2 consists of the crowded pedestrian walking horizontally to the camera plane. The examples of the UCSD are shown in Fig. 16.17, where (a) is Ped1 and (b) is Ped2.

16.4.1.2 UMN dataset

The UMN dataset [1] is one of the publicly available benchmarks in the video anomaly detection task designed for identifying the anomalies in crowds. It contains 11 videos with 7700 frames recorded in various indoor and outdoor scenarios. All frames have a resolution of 320×240 pixels. Both indoor and outdoor scenes have the characteristics

Fig. 16.17 UCSD pedestrian dataset: (A) Ped1 and (B) Ped2.

Fig. 16.18 UMN dataset.

of walking pedestrian as the normal event and the running pedestrian as the abnormal event as shown in Fig. 16.18. All video sequences start with the walking patterns then end with the running patterns.

16.4.1.3 CHUK Avenue dataset

The CUHK Avenue dataset [6] consists of the crowded scenes at the campus. The total number of frames is 30,652 frames consisting of 15,328 frames for 16 training videos and 15,324 frames for 21 test videos. Each video sequence has a length of 1–2 min, with 25 frames per *sec*ond (fps). This dataset is challenging due to its various moving objects in crowds and types of anomaly patterns related to human actions, including object-related human actions (person throwing, grabbing, and leaving objects), running, jumping, and loitering. In contrast, the normal pattern is the crowds who walk parallel to the image plane. The examples of the CHUK Avenue dataset are shown in Fig. 16.19.

16.4.2 Implementation details

We implement the proposed DSTN framework using Keras [51] machine learning platform with TensorFlow [52] backend, and Matlab. A GPU NVIDIA GeForce GTX, 1080 Ti with 3584 CUDA Cores and 484 GB/sec memory bandwidth, is used during the training procedure. Additionally, the testing measurement is employed on a CPU Intel Core i9-7960 × with a 2.80 GHz processor base frequency. The model performs transformation learning from spatial to temporal representations with the help of Adam optimizer. A learning rate is set to 0.0002, while the exponential decay rate β_1 and β_2 are set to 0.9 and 0.999, respectively, with epsilon 10^{-8}.

Fig. 16.19 CHUK Avenue dataset.

16.4.3 Evaluation criteria

16.4.3.1 Receiver operating characteristic (ROC)

The Receiver operation characteristic curve (ROC) is a standard method used for evaluating the performance of an anomaly detection system. It is a plot that indicates a comparison between true positive rate (*TPR*) and false positive rate (*FPR*) at various threshold criteria [53] and benefits the analysis of the decision-making process.

In the anomaly detection observation, the abnormal events that are correctly determined as the positive detections (abnormal event) from the entire positive ground truth data are represented as *TPR* known as the probability of detection. The more the curve of *TPR* goes up, the better the detection accuracy of abnormal events is. The normal event (negative data) that are incorrectly determined as the positive detections from the entire negative ground truth data are represented as *FPR*. The higher *FPR* means the higher rate of the misclassification of normal events. There are four types of binary predictions for *TPR* and *FPR* computation, as described below.

True positive (*TP*) is the correct positive detection of an abnormal event when the prediction outcome and the ground truth data are positive (abnormal event).

False positive (*FP*) is the false positive detection when the outcome is predicted as positive (abnormal event), but the ground truth data is negative (normal event), meaning that the normal event is incorrectly detected as an abnormal event. This problem often occurs in the video anomaly detection task (e.g., a walking person is detected as an anomaly).

True negative (*TN*) is the correct detection of a normal event when the outcome is predicted as negative (normal event) and the ground truth data is also negative.

False negative (*FN*) is the incorrect detection when the outcome is predicted as negative (normal event) and the ground truth data is positive (abnormal event).

Hence, *TPR* and *FPR* can be computed, as shown in Eqs. (16.32) and (16.33), respectively:

$$TPR = TP/(TP + FN) \tag{16.32}$$

$$FPR = FP/(FP + TN) \tag{16.33}$$

16.4.3.2 Area under curve (AUC)

Area Under Curve, also known as AUC, is used in classification analysis problems to define the best prediction model. It is computed from all the areas under the ROC curve, where *TPR* is plotted against *FPR*. The higher value of AUC indicates the superior performance of the model. Ideally, the model is a perfect classifier when all positive data are ranked above all negative data (AUC = 1). In practice, most of the AUC results are required in the range between 0.5 and 1.0 (AUC = [0.5,1]), meaning that the random positive data are ranked higher than the random negative data (greater than 50%). Besides, the worst case is when all negative data are ranked above all positive data, leading

the AUC to 0 (AUC = 0). Hence, AUC classifiers can be defined as AUC $\in$ [0,1] where AUC values for real-world use are greater than 0.5. The AUC values that are less than 0.5 are not acceptable for the model [53]. To conclude, we prefer higher AUC values than the lower ones.

16.4.3.3 Equal error rate (EER)

Apart from the AUC, the performance of the model can be quantified by observing a receiving operating characteristic equal error rate (ROC-EER). The EER is a fixpoint that specifies equality of the misclassification of positive and negative data. Specifically, EER can be obtained from the intersection of the ROC curve on the diagonal EER line by varying a threshold until *FPR* equals to the miss rate 1-*TPR*. The lower EER values indicate that the model has better performance.

16.4.3.4 Frame-level and pixel-level evaluations for anomaly detection

In general, the quantitative performance evaluation of the anomaly detection consists of two criteria, including frame-level and pixel-level evaluations. The frame-level evaluation focuses on the detection rate of the anomalous event in the scene. If one or more anomalous pixels are detected, the frame will be labeled as the abnormal frame no matter what size and location of the abnormal objects are. In this case, the detected frame is defined as *TP* if the actual frame is also abnormal. Contrarily, if the actual frame is normal, then the detected frame is classified as *FP*. The evaluation of the pixel level determines the correct location of anomalous object detection in the scene. This evaluation is a challenging criterion in anomaly detection and localization research since it focuses on the local pixel. It is remarkably more demanding and stricter than the frame-level evaluation due to its complexity of localizing anomalies, which improves the accuracy of the frame-level anomaly detection. To indicate whether the frame is the true positive (*TP*), the detected abnormal area is needed to be overlapped more than 40% with the ground truth [3]. In addition, the frame will be distinguished as the false positive (*FP*) if one pixel is detected as abnormal events.

16.4.3.5 Pixel accuracy

In a standard semantic segmentation evaluation, the pixel accuracy metric [54] is computed to define the correctness of the pixel belonging to each semantic class. In the proposed DSTN, two semantic classes are defined, including a foreground semantic class and a background semantic class. The pixel accuracy is defined as $\sum_i n_{ii} / \sum_i n_{ti}$, where n_{ij} is the number of misclassified pixels of class i, and n_{ti} is the total of consisting pixels of class i.

16.4.3.6 Structural similarity index (SSIM)

SSIM index is a perceptual metric to measure the image quality of the predicted image to its original image [55]. Using the SSIM index, the model is more effective when the

predicted image is more similar to the target image. In our case, we use SSIM to analyze the similarity of the dense optical flow generated from the generator to its real dense optical flow obtained from two consecutive video frames.

16.4.4 Performance of DSTN

We evaluate the proposed DSTN regarding accuracy and time complexity aspects. The ROC curve is used to illustrate the performance of anomaly detection at the frame level and the pixel level and analyze the experimental results with other state-of-the-art works. Additionally, the AUC and the EER are evaluated as the criteria for determining the results.

The performance of DSTN is first evaluated on the UCSD dataset, consisting of 10 and 12 videos for the UCSD Ped1 and the UCSD Ped2 with the pixel-level ground truth, by using both frame-level and pixel-level protocols. In the first stage of DSTN, patch extraction is implemented to provide the appearance features of the foreground object and its motion regarding the vector changes in each patch. The patches are extracted independently from each original image with a size of 238×158 pixels (UCSD Ped1) and 360×240 pixels (UCSD Ped2) to apply it with $(w/4) \times h \times c_p$. As a result, we obtain 22 k patches from the UCSD Ped1 and 13.6 k patches from the UCSD Ped2. Then, to feed into the spatiotemporal translation model, we resize all patches to the 256×256 default size in both training and testing time.

During training, the input of G (the concatenation of f and f_{BR} patches) and target data (the generated dense optical flow OF_{gen}) are set to the same size as the default resolution of 256×256 pixels. The encoding and decoding modules in G are implemented differently. As in the encoder network, the image resolution is encoded from $256 \rightarrow 128 \rightarrow 64 \rightarrow 32 \rightarrow 16 \rightarrow 8 \rightarrow 4 \rightarrow 2 \rightarrow 1$ to obtain the latent space representing the spatial image in one-dimensional data space. CNN effectively employs this downscale with kernels of 3×3 and stride $s = 2$. Additionally, the number of neurons corresponding to the image resolution in En is introduced in each layer from $6 \rightarrow 64 \rightarrow 128 \rightarrow 256 \rightarrow 512 \rightarrow 512 \rightarrow 512 \rightarrow 512 \rightarrow 512$. In contrast, De decodes the latent space to reach the target data (the temporal representation of OF_{gen}) with a size of 256×256 pixels using the same structure as En. The dropout is employed in De as noise z to remove the neuron connections using probability $p = 0.5$, resulting in the prevention of overfitting on the training samples. Since D needs to fluctuate G to correct the classification between real and fake images at training time, PatchGAN is then applied by inputting a patch size of 64×64 pixels to output the probability of class label for the object. The PatchGAN architecture is constructed from $64 \rightarrow 32 \rightarrow 16 \rightarrow 8 \rightarrow 4 \rightarrow 2 \rightarrow 1$, which is then flattened to 512 neurons and plugged in with Fully Connection (FC) and Softmax layers. The use of PatchGAN benefits the model in terms of time complexity. This is probably because there are fewer parameters to learn on the partial image, making

the model less complex and can achieve good running time for the training process. In the aspect of testing, G is specifically employed to reconstruct OF_{gen} in order to analyze the real motion information OF_{fus}. The image resolution for testing and training are set to the same resolution for all datasets.

The quantitative performance of DSTN is presented in Table 16.1 where we consider the DSTN with various state-of-the-art works, e.g., AMDN [15], GMM-FCN [12], Convolutional AE [14], and future frame prediction [21]. From Table 16.1 it can be observed that the DSTN overcomes most of the methods in both frame-level and pixel-level criteria since we achieve higher AUC and lower EER on the UCSD Dataset. Moreover, we show the qualitative performance of DSTN using the standard evaluation for anomaly detection research known as the ROC curves, where we vary a threshold from 0 to 1 to plot the curve of *TPR* against *FPR*. The qualitative performance of DSTN is compared with other approaches in both frame-level evaluation (see Fig. 16.20A) and pixel-level evaluation (see Fig. 16.20B) on the UCSD Ped1 and at the frame-level evaluation on the UCSD Ped2 as presented in Figs. 16.20 and 16.21, respectively. Following Figs. 16.20 and 16.21, the DSTN (circle) shows the strongest growth curve on *TPR* and overcomes all the competing methods in the frame and pixel level. This means that the DSTN is a reliable and effective method to be able to detect and localize the anomalies with high precision.

Examples of the experimental results of DSTN on the UCSD Ped 1 and Ped 2 dataset are illustrated in Fig. 16.22 to extensively present its performance in detecting and localizing anomalies in the scene. According to Fig. 16.22, the proposed DSTN is able to detect and locate various types of abnormalities effectively with each object, e.g., (a) a wheelchair, (b) a vehicle, (c) a skateboard, and (d) a bicycle, or even more than one anomaly in the same scene, e.g., (e) bicycles, (f) a vehicle and a bicycle, and (g) a bicycle and a skateboard. However, we face the false positive problems in Fig. 16.22H a bicycle and a skateboard, where the walking person (normal event) is detected as an anomaly. Even the bicycle and the skateboard are correctly detected as anomalies in Fig. 16.22H, the false detection on the walking person makes this frame wrong anyway. The false positive anomaly detection is probably caused by a similar speed of walking to cycling in the scene.

For the UMN dataset, the performance of DSTN is evaluated using the same settings as training parameters and network configuration on the UCSD pedestrian dataset. Table 16.2 indicates the AUC performance comparison of the DSTN with various competing works such as GANs [20], adversarial discriminator [22], AnomalyNet [23], and so on. Table 16.2 shows that the proposed DSTN achieves the best AUC results the same as Ref. [23], which outperforms all other methods. Noticeably, most of the competing methods can achieve high AUC on the UMN dataset. This is because the UMN dataset has less complexity regarding its abnormal patterns than the UCSD pedestrian and the Avenue datasets. Fig. 16.23 shows the performance of DSTN in detecting and localizing

Table 16.1 EER and AUC comparison of DSTN with other methods on UCSD Dataset [26].

	Ped1 (frame level)		Ped1 (pixel level)		Ped2 (frame level)		Ped2 (pixel level)	
Method	**EER**	**AUC**	**EER**	**AUC**	**EER**	**AUC**	**EER**	**AUC**
MPPCA	40%	59.0%	81%	20.5%	30%	69.3%	–	–
Social Force (SF)	31%	67.5%	79%	19.7%	42%	55.6%	80%	–
SF + MPPCA	32%	68.8%	71%	21.3%	36%	61.3%	72%	–
Sparse Reconstruction	19%	–	54%	45.3%	–	–	–	–
MDT	25%	81.8%	58%	44.1%	25%	82.9%	54%	–
Detection at 150fps	15%	91.8%	43%	63.8%	–	–	–	–
SR + VAE	16%	90.2%	41.6%	64.1%	18%	89.1%	–	–
AMDN (double fusion)	16%	92.1%	40.1%	67.2%	17%	90.8%	–	–
GMM	15.1%	92.5%	35.1%	69.9%	–	–	–	–
Plug-and-Play CNN	8%	95.7%	40.8%	64.5%	18%	88.4%	–	–
GANs	8%	97.4%	35%	70.3%	14%	93.5%	–	–
GMM-FCN	11.3%	94.9%	36.3%	71.4%	12.6%	92.2%	**19.2%**	78.2%
Convolutional AE	27.9%	81%	–	–	21.7%	90%	–	–
Liu et al.	23.5%	83.1%	–	33.4%	12%	95.4%	–	40.6%
Adversarial discriminator	7%	96.8%	34%	70.8%	11%	**95.5%**	–	–
AnomalyNet	25.2%	83.5%	–	45.2%	10.3%	94.9%	–	52.8%
DSTN (proposed method)	**5.2%**	**98.5%**	**27.3%**	**77.4%**	**9.4%**	**95.5%**	21.8%	**83.1%**

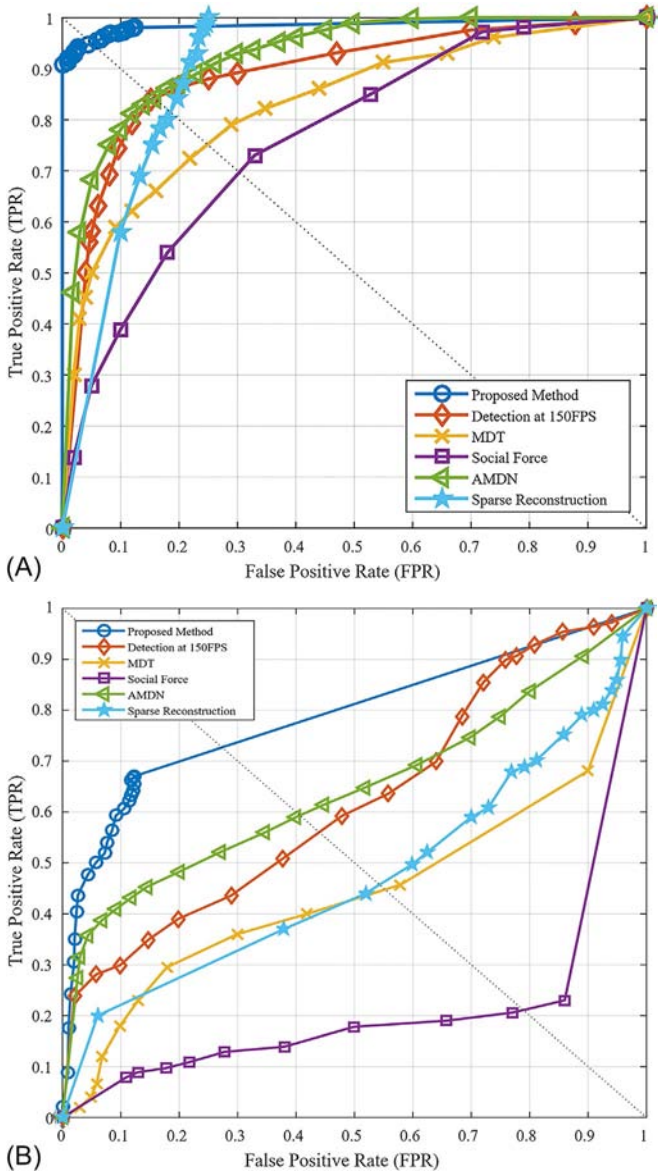

Fig. 16.20 ROC Comparison of DSTN with other methods on UCSD Ped1 dataset: (A) frame-level evaluation and (B) pixel-level evaluation [26].

anomalies in different scenarios on the UMN dataset, including (d) an indoor and outdoors in (a), (b), and (c), where we can detect most of the individual objects in the crowded scene.

Apart from evaluating DSTN on the UCSD and the UMN datasets, we also assess our performance on the challenging CUHK Avenue dataset with the same parameter and

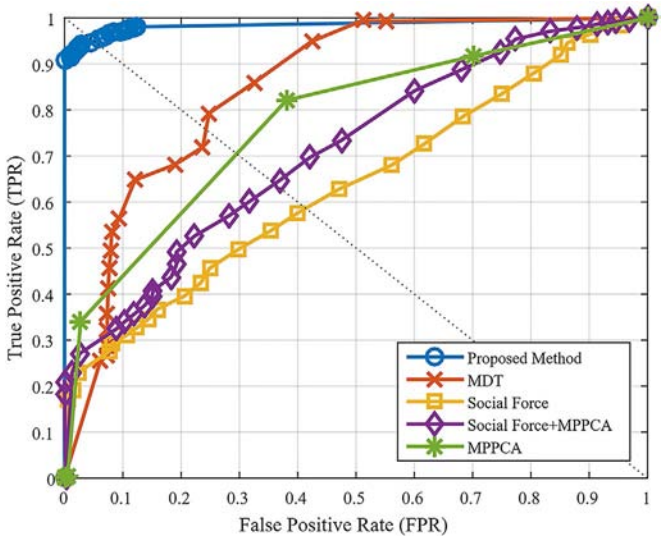

Fig. 16.21 ROC Comparison of DSTN with other methods on UCSD Ped2 dataset at frame-level evaluation [26].

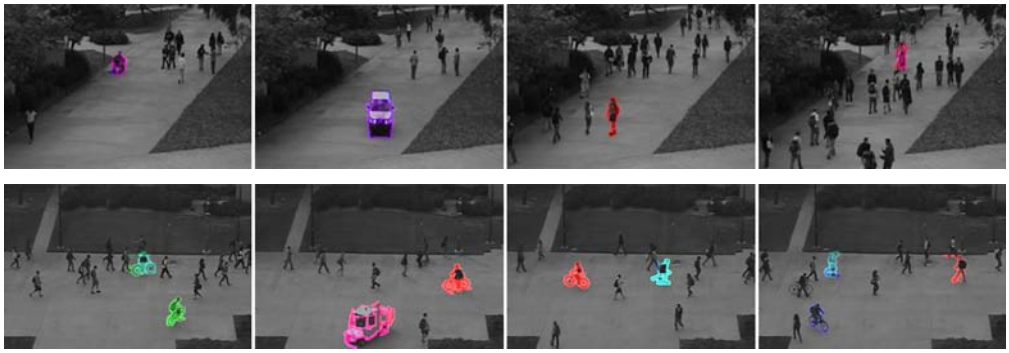

Fig. 16.22 Examples of DSTN performance in detecting on localizing anomalies on UCSD Ped1 and Ped2 dataset: (A) a wheelchair, (B) a vehicle, (C) a skateboard, (D) a bicycle, (E) bicycles, (F) a vehicle and a bicycle, (G) a bicycle and a skateboard, and (H) a bicycle and a skateboard [26].

configuration settings as the UCSD and the UMN datasets. Table 16.3 presents the performance comparison in terms of EER and AUC of the DSTN with other competing works [6, 12, 14, 21, 23] in which the proposed DSTN surpasses all state-of-the-art works for both protocols. We show examples of the DSTN performance in detecting and localizing various types of anomalies, e.g., (a) jumping, (b) throwing papers, (c) falling papers, and (d) grabbing a bag, on the CUHK Avenue dataset in Fig. 16.24 The DSTN can effectively detect and localize anomalies in this dataset, even in

Table 16.2 AUC comparison of DSTN with other methods on UMN dataset [26].

Method	AUC
Optical-flow	0.84
SFM	0.96
Sparse reconstruction	0.976
Commotion	0.988
Plug-and-play CNN	0.988
GANs	0.99
Adversarial discriminator	0.99
Anomalynet	**0.996**
DSTN (proposed method)	**0.996**

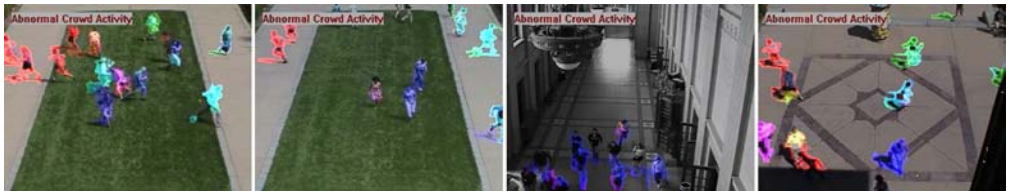

Fig. 16.23 Examples of DSTN performance in detecting on localizing anomalies on UMN dataset, where (A), (B), and (D) contain running activity outdoors while (C) is in an indoor [26].

Table 16.3 EER and AUC comparison of DSTN with other methods on CUHK Avenue dataset [26].

Method	EER	AUC
Convolutional AE	25.1%	70.2%
Detection at 150 FPS	–	80.9%
GMM-FCN	22.7%	83.4%
Liu et al.	–	85.1%
Anomalynet	22%	86.1%
DSTN (proposed method)	**20.2%**	**87.9%**

Fig. 16.24D, which contains only small movements for abnormal events (only the human head and the fallen bag are slightly moving).

To indicate the significance of our performance for real-time use, we then compare the running time of DSTN during testing in seconds per frame as shown in Table 16.4 with other competing methods [3–6, 15] following the environment and the computational time from Ref. [15].

Regarding Table 16.4, we achieve a lower running time than most of the competing methods except for Ref. [6]. This is because the architecture of DSTN relies on the

Fig. 16.24 Examples of DSTN performance in detecting on localizing anomalies on CUHK Avenue dataset: (A) jumping, (B) throwing papers, (C) falling papers, and (D) grabbing a bag [26].

Table 16.4 Running time comparison on testing measurement (seconds per frame).

Method	CPU (GHz)	GPU	Memory (GB)	Running time			
				Ped1	Ped2	UMN	Avenue
Sparse Reconstruction	2.6	–	2.0	3.8	–	0.8	–
Detection at 150 fps	3.4	–	8.0	**0.007**	–	–	**0.007**
MDT	3.9	–	2.0	17	23	–	–
Li et al.	2.8	–	2.0	0.65	0.80	–	–
AMDN (double fusion)	2.1	Nvidia Quadro K4000	32	5.2	–	–	–
DSTN (proposed method)	2.8	–	24	0.315	**0.319**	**0.318**	0.334

framework of the deep learning model using multiple layers of the convolutional neural network, which is more complex than Ref. [6] that uses the learning of a sparse dictionary and provides fewer connections. However, according to the experimental results in Tables 16.1 and 16.3, our proposed DSTN significantly provides higher AUC and lower EER with respect to the frame and the pixel level on the CUHK Avenue and the UCSD pedestrian datasets than Ref. [6]. Regarding the running time, the proposed method runs 3.17 fps for the UCSD Ped1 dataset, 3.15 fps for the UCSD Ped2 dataset, 3.15 fps for the UMN dataset, and 3 fps for the CUHK Avenue dataset. In every respect, we provide the performance comparison of the proposed DSTN with other competing works [3–6, 15] to show our performance in regard to the frame-level AUC and the running time in seconds per frame for the UCSD Ped1 and Ped2 dataset as presented in Figs. 16.25 and 16.26, respectively.

Considering Figs. 16.25 and 16.26, our proposed method achieves the best results regarding the AUC and running time aspects. In this way, we can conclude that our DSTN surpasses other state-of-the-art approaches since we reach the highest AUC values at the frame level anomaly detection and the pixel level localization and given the good computational time for real-world applications.

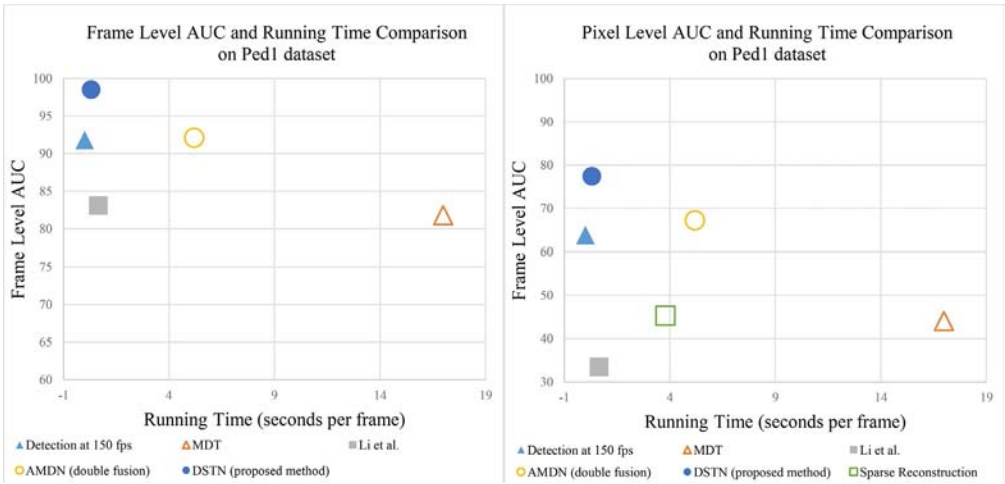

Fig. 16.25 Frame-level AUC comparison and running time on UCSD Ped1 dataset.

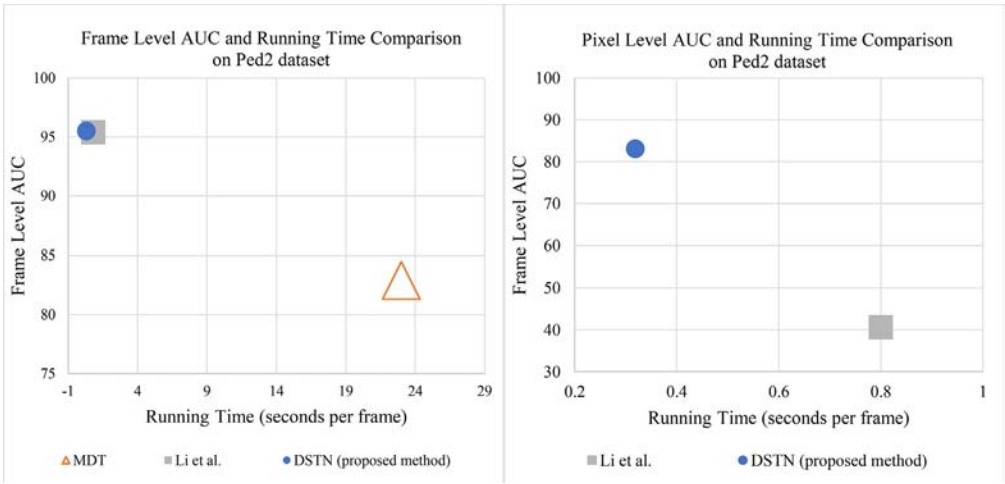

Fig. 16.26 Frame-level AUC comparison and running time on UCSD Ped2 dataset.

16.4.5 The comparison of generative adversarial network with an autoencoder

GAN-based U-Net architecture is a practical approach to shortcut low-level information across the network. The skip connections in the generator play a significant role in our proposed framework. We highlight its significance with the experiments on the UCSD Ped2 and compare it with autoencoder, which can be constructed by erasing the skip connections in the U-Net architecture. All training videos are learned on both skip

connections and autoencoder for 40 epochs to observe the performance in minimizing the L1 loss, as shown in Fig. 16.27. From Fig. 16.27, it demonstrates that the loss curve of the skip connections reaches lower error over the training time than the loss curve of the autoencoder, showing superior performance of the skip connections over the autoencoder.

Besides, we observe the ability to generate temporal information (generated dense optical flow) of the skip connections and the autoencoder using the test videos from the UCSD Ped2 and compare it to the dense optical flow ground truth as displayed in Fig. 16.28. The autoencoder is impotent to achieve the motion information shown in Fig. 16.28C. In contrast, the skip connections in Fig. 16.28B can produce the motion

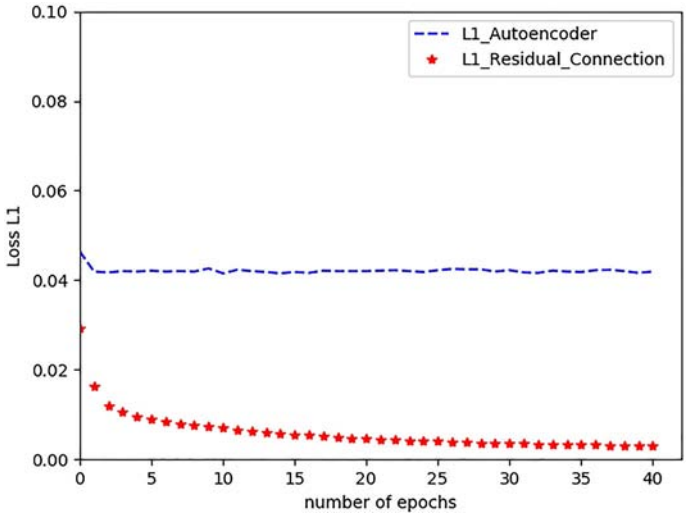

Fig. 16.27 Performance comparison on UCSD Ped2 dataset between GAN based U-Net architecture (the residual connection) and autoencoder [26].

Fig. 16.28 The qualitative results in generating (A) dense optical flow on UCSD Ped2 dataset between (B) residual connection and (C) autoencoder [26].

Table 16.5 FCN-score and SSIM comparison on UCSD Ped2 dataset between residual connection and autoencoder.

Network architecture	Pixel accuracy	SSIM
Autoencoder	0.83	0.82
Residual connection	**0.9**	**0.96**

information of dense optical flow that correctly corresponds to its ground truth in Fig. 16.28A, giving a good synthesized image quality.

To indicate the quantitative performance of the skip connections and the autoencoder, Structural Similarity Index (SSIM) [55] and FCN-score [54] are evaluated for each architecture on the UCSD Ped2 as presented in Table 16.5 A higher value means better performance for both evaluation criteria.

Table 16.5 shows that the GAN-based U-Net architecture with skip connection is more suitable for the low-level information since it achieves superior results than the autoencoder for both evaluation metrics, especially in the SSIM.

16.4.6 Advantages and limitations of generative adversarial network for video anomaly detection

The generative adversarial networks for anomaly detection have certain advantages over the traditional CNNs. The advantages are that the GAN framework does not require any labeled data and inference during the learning procedure. In addition, GAN can generate the example data without using different entries in a sequential sample and does not need Markov Chain Monte Carlo (MCMC) method to train the model as the Adversarially trained AAE [24] and VAE [25]. Instead, it computes only the backpropagation to obtain the gradients. As regards statistical advantage, the GAN model may gain the density distribution of example data from the generator network, which is trained and updated with the gradients flowing through the discriminator rather than directly updating with the example data. In this way, for GAN in video anomaly detection task, the objective function of the generator is strengthened to be able to generate the synthetic output that looks real from the input image since the parameters of the generator do not directly obtain the components of the target image. Apart from the advantages mentioned above, the generator network provides a very sharp synthetic image, while the visual performance of the VAE network based on the MCMC method presents a blurry image for mixing the modes in chains. As in the limitations of GAN, the training of GAN is unstable compared to VAE, resulting in the difficulty of predicting the value of each pixel for the whole image and causing artifact noise in the synthetic image. The major limitation of anomaly detection using GAN in the current research trends is that only the static camera scenario is implemented to obtain the appearance and motion features from the moving foreground objects. Besides, GAN also has a problem in learning and generating small objects

(full appearance of the objects) in the crowded scene, making it challenging to enhance the accuracy of the model, especially at the pixel level.

16.5 Summary

In this chapter, we extensively explain the architecture of GANs and explore its applications on video anomaly detection research. DSTN, a novel unsupervised anomaly detection and localization method, is introduced to enhance the knowledge of GAN and improve the performance of the system with respect to the accuracy of anomaly detection at the frame level and localization at the pixel level and the computational time. The DSTN is intended to comprehensively master the features from the spatial to the temporal representations by employing the novel fusion between the background removal and the real dense optical flow. The concatenation of patches is presented to assist the learning of the generative network. The proposed method is an unsupervised manner since only the normalities are trained to obtain the corresponding generated dense optical flow without labeling abnormal data. Since all videos are input into the model during testing, the unrecognized patterns are classified as abnormalities because the model has no prior knowledge of any abnormal events. The abnormalities can be simply detected by subtracting the difference in local pixels between the real and the generated dense optical flow images. To the best of our knowledge, the proposed DSTN is the first attempt to boost pixel-level anomaly localization with the edge wrapping method as the postprocessing process of the GAN framework. We implemented three publicly available benchmarks; UCSD pedestrian, UMN, and CUHK Avenue datasets. The performance of DSTN is distinguished with various methods and analyzed with the autoencoder to specify the significance of using the skip connections of GAN. From the experimental results, the proposed DSTN outperforms other state-of-the-art works for anomaly detection and localization and time consumption. The advantages and limitations of GAN are addressed in the final section to deliver a comprehensive view of the use of GAN for the video anomaly detection task.

References

[1] R. Mehran, A. Oyama, M. Shah, Abnormal crowd behavior detection using social force model, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2009, pp. 935–942.

[2] J. Kim, K. Grauman, Observe locally, infer globally: a space-time MRF for detecting abnormal activities with incremental updates, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2009, pp. 2921–2928.

[3] W. Li, V. Mahadevan, N. Vasconcelos, Anomaly detection and localization in crowded scenes, IEEE Trans. Pattern Anal. Mach. Intell. 36 (2013) 18–32.

[4] V. Mahadevan, W. Li, V. Bhalodia, N. Vasconcelos, Anomaly detection in crowded scenes, in: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE, 2010, pp. 1975–1981.