TWO-WAY MULTI-INPUT GENERATIVE NEURAL NETWORK FOR ANOMALY EVENT DETECTION AND LOCALIZATION

TWO-WAY MULTI-INPUT GENERATIVE NEURAL NETWORK FOR ANOMALY EVENT DETECTION AND LOCALIZATION

By MINGCHEN YANG, B.ENG

A Thesis Submitted to the School of Graduate Studies in Partial Fulfilment of the Requirements for the Degree Master of Applied Science

McMaster University

MASTER OF APPLIED SCIENCE (2022)

Hamilton, Ontario (Electrical and Computer Engineering)

TITLE: Two-way Multi-input Generative Neural Network for Anomaly Event Detection and Localization

AUTHOR: Mingchen Yang, B.Eng (Xidian University)

SUPERVISOR: Dr. Shahram Shirani

NUMBER OF PAGES: xvii, 70

Lay Abstract

Recently, abnormal event detection has attracted increasing attention in the field of surveillance video. However, it is still a big challenge to build an automatic and reliable abnormal event detection system to review a surveillance video containing hundreds of frames and mask the frames with abnormal objects or events. In this thesis, we build a model and teach it to memorize the structure of normal frames. Then the model is able to tell which frames are normal. Any other frames that appear in the surveillance video will be classified as abnormal frames. Moreover, we design a new method to evaluate the performance of our model and compare it with other models' results.

Abstract

Anomaly event detection has become increasingly important and is of great significance for real-time monitoring systems. However, developing a reliable anomaly detection and localization model still requires overcoming many challenging problems considering the ambiguity in the definition of an abnormal event and the lack of ground truth datasets for training. In this thesis, we propose a Two-way Multi-input Generative Neural Network (TMGNN), which is an unsupervised anomaly events detection and localization method based on Generative Adversarial Network (GAN). TMGNN is composed of two neural networks, an appearance generation neural network and a motion generation neural network. These two networks are trained on normal frames and their corresponding motion and mosaic frames respectively. In the testing steps, the trained model cannot properly reconstruct the anomalous objects since the network is trained only on normal frames and has not learned patterns of anomalous cases. With the help of our new patch-based evaluation method, we utilize the reconstruction error to detect and localize possible anomalous objects. Our experiments show that on the UCSD Pedestrain2 dataset, our approach achieves 96.5% Area Under Curve (AUC) and 94.1% AUC for the frame-level and pixel-level criteria, respectively, reaching the best classification results

compared to other traditional and deep learning methods.

Acknowledgements

I would like to express my deepest appreciation to my supervisor, Dr. Shahram Shirani, for his valuable suggestions, enthusiastic encouragement and patient guidance throughout the duration of my graduate studies. Without his plentiful experience and unwavering support, this thesis would have never been accomplished. He always provided helpful and constructive advice when I encountered difficulties in my research. It's my honor and luck to be his student.

I would also like to extend my sincere thanks to the committee members, Dr. Jun Chen and Dr. Sorina Dumitrescu, for reading this thesis and giving helpful advice for my future work.

Many thanks to my family for their continuous love and support throughout my life. Thanks for giving me the strength and courage to chase my dreams.

To my friends, thank you for the encouragement and the joy you have brought me. Your friendship makes my daily life an unforgettable experience.

Contents

List of Figures and Tables

List of Abbreviations

HOG	Histograms of Oriented Gradients
HOS	Histograms of Oriented Swarms
ConvLSTM	Convolutional Long Short-term Memory
GAN	Generative Adversarial Network
TMGNN	Two-way Multi-input Generative Neural Network
ANN	Artificial Neural Network
MLP	Multi-layer Perceptron
CNN	Convolutional Neural Network
ReLU	Rectified Linear Unit
KL divergence	Kullback-Leibler Divergence
CGAN	Conditional Generative Adversarial Network
DCGAN	Deep Convolutional Generative Adversarial Network

WGAN	Wasserstein Generative Adversarial Network
LDA	Linear Discriminant Analysis
AdaBoost	Adaptive Boosting
SSIM	Structural Similarity Index Measure
ROC	Receiver Operating Characteristic Curve
AUC	Area Under the Receiver Operating Characteristic Curve
EER	Equal Error Rate
TPR	True Positive Rate
FPR	False Positive Rate

List of Symbols

G	The generator of GAN
D	The discriminator of GAN
z	The random noise
x	The real sample
$G(z)$	The output of the generator
X	The set of x
$V(G,D)$	The cross-entropy function of a binary classifier
$D(x)$	The output of the discriminator when the input of the discriminator is x
$D(G(z))$	The output of the discriminator when the input is $G(z)$
N^{appe}	The appearance generation neural network
N^{motion}	The motion generation neural network
$P(A,B)$	The probability at which A and B occur together

$P(A \mid B)$	The probability of A under the condition B
$P(B \mid A)$	The probability of B under the condition A
$P(A)$	The probability of A
F_t	The t-th original image
K_{24}	The set of 24 kernels
R_t	The t-th residual map
Mt_t	The t-th motion image
G^{appe}	The generator of N^{appe}
G^{motion}	The generator of N^{motion}
Ms_t	The t-th mosaic image
I_t^{appe}	The concatenation of Ms_t and Mt_t
P_t^{appe}	The t-th output of G^{appe}
P_t^{motion}	The t-th output of G^{motion}
D^{appe}	The discriminator of N^{appe}
D^{motion}	The discriminator of N^{motion}

S_t^{appe}	The *t*-th output of D^{appe}
S_t^{motion}	The *t*-th output of D^{motion}
l	The luminance of an image
c	The contrast of an image
s	The structure of an image
c_1, c_2, c_3	The constants of three SSIM components
α, β, γ	The weights of three SSIM components
μ_I, μ_J	The mean value of image *I* and image *J* respectively
σ_I, σ_J	The variance of image *I* and image *J* respectively
$\sigma_{I,J}$	The covariance of image *I* and image *J*
$\lambda_1, \lambda_2, \lambda_3$	The weights of different loss functions
Re_t	The *t*-th reconstruction image
Or_t	The *t*-th ground truth image
$Diff_t$	The absolute difference between Re_t and Or_t
$normDiff_t$	The normalized $Diff_t$

$Thres_{val}$	The pixel intensity threshold of the evaluation method
$Thres_{num}$	The number threshold of the evaluation method
E_t	The square error between Re_t and Or_t
S_t	The regularity score of the pixel-based evaluation method
$Diff_t^{appe}$	The absolute difference between P_t^{appe} and F_t
$Diff_t^{motion}$	The absolute difference between P_t^{motion} and Mt_t
V	The video clip in the training and testing dataset
m_v^{appe}	The maximum pixel value of $Diff_t^{appe}$ in the whole video clip
m_v^{motion}	The maximum pixel value of $Diff_t^{motion}$ in the whole video clip
$normDiff_t^{appe}$	The normalized $Diff_t^{appe}$
$normDiff_t^{motion}$	The normalized $Diff_t^{motion}$
η_1, η_2	The weights to control the ratio between $normDiff_t^{appe}$ and $normDiff_t^{motion}$
N_{ab}	The number of abnormal patches of an image

1 Introduction

1.1 Current Challenge

Over the past 20 years, as technology has evolved and computer vision algorithms have improved, real-time surveillance systems have been installed extensively to protect personal safety and property. Many companies and schools are interested in having a video surveillance system on their premises and installing a surveillance camera at residential properties has become popular as well.

It is a waste of time and labor to go through the entire surveillance video to locate anomalies. More importantly, it would be helpful if we can determine which frame contains anomaly objects automatically in real-time. Therefore, many researchers and scientists are working on the design of an automatic real-time anomaly event detection system based on deep learning.

There are still some critical challenges to be addressed in the area of anomaly event detection. The most important is our inability to collect enough abnormal videos or images as our ground truth dataset. Moreover, there is no consistent and objective definition of what an anomalous event is. Abnormal objects differ depending on the

context. An object or event can have different attributes (normal or abnormal) in different environments. For example, riding a motorcycle on a sidewalk is abnormal but riding it on the road is normal. Also, it is much easier to collect enough normal videos or frames in our daily life, since the number of abnormal videos is much less than the number of normal videos.

Nowadays, most anomaly event detection methods could be broadly grouped into three categories: (1) object trajectory methods [1][2], (2) handcraft methods [3][4][5], and (3) deep learning methods [6][7][8]. The main idea of the object trajectory method is modelling the normal trajectory of all objects on each image and learning the specific parameters of object trajectories. Any object with an abnormal trajectory will be detected as an anomalous event. However, this method requires computing the trajectories of all objects, which is usually time-consuming and computationally expensive. It is hard to deploy these methods in crowded scenes with multiple objects. Handcraft method is more robust and efficient. A scene descriptor is constructed from optical flow frames [4], Histograms of Oriented Gradients (HOG) or Histograms of Oriented Swarms (HOS) [3]. The scene descriptor learns a distribution of normal behaviors that allows the method to identify abnormal events. Compared with the object trajectory method, the handcraft method is computationally less expensive and performs better in anomalous event detection tasks for complex and crowded scenes because the scene descriptors reflect

more high-level information about the objects and their motion characteristics. Nevertheless, the handcraft method relies on appropriate and specific scene descriptors and it is not able to be extended to other anomaly event detection tasks.

Convolutional neural networks have been widely used in natural language processing and computer vision areas, such as speech recognition [9][10] and super resolution [11][12], in the past 20 years. Some recent works have attempted to take advantage of convolutional neural networks to tackle the anomaly event detection problem. Generative adversarial networks have been proposed in [13] and are frequently applied to improve the performance of anomaly event detection algorithms due to their powerful reconstruction capabilities. It consists of two neural networks, a generator (G) and a discriminator (D). The generator is designed to generate "fake" images to deceive the discriminator, while the discriminator acts as a supervisor to select these "fake" images from a large number of "real" images. In the training process, more and more realistic images are generated from the generator, and the discriminator is optimized to identify which inputs are "fake" images and which are "real" data.

Even though generative adversarial networks have shown many advantages in building a generic anomaly event detection system, there are still some challenges in training a perfect generator and discriminator. Since the training process for generators is much more difficult than the discriminator's training process, we have to make sure that

the generator is synchronized well with the discriminator in order to avoid “the Helvetica scenario” [13]. If the discriminator is more powerful than the generator, it will be able to distinguish all generated frames, and the neurons inside the generator will stop learning any reconstruction patterns and vice versa. Many recent works are still struggling to find the best way to maintain the balance between the generator and discriminator.

[8] proposed a bidirectional convolutional long short-term memory (ConvLSTM) [14] with a generative adversarial network (GAN) to predict the future frame and utilized the reconstruction errors between predicted frames and ground truth frames to identify which frame is anomalous. [6] built a FlowNet based on GAN and used the optical flow images and video prediction framework to detect anomaly events.

1.2 Thesis Outline

This thesis consists of 5 chapters. Chapter 2 provides the background of convolutional neural networks and generative adversarial networks and introduces two traditional anomaly event detection methods and one deep learning method. Chapter 3 represents the architecture of our proposed two-way multi-input generative neural network, the details of the patch-based evaluation method and the loss functions used in the training steps. Chapter 4 introduces a few evaluation metrics designed to evaluate the performance of

classification models and shows the experimental results at the frame-level and pixel-level criteria. Chapter 5 demonstrates the conclusion of the thesis and future works.

1.3 Thesis Contributions

In this thesis, we propose a two-way multi-input generative neural network (TMGNN) for anomaly event detection and localization tasks and a new patch-based evaluation method. TMGNN includes two neural networks: (1) an appearance generation neural network and (2) a motion generation neural network. In the training steps, only normal frames are fed into the two neural networks. During testing, since the model has only learned the patterns of normal objects, it cannot reconstruct abnormal objects and fails to generate abnormal frames. We compute the absolute difference between reconstructions and ground truth images and utilize this reconstruction error to detect anomalous events. To train a balanced generative adversarial network, we adopt multiple inputs to the appearance generation neural network, which provides the generator with sufficient grayscale and shape information to improve the ability of generating more realistic-looking images. Experiments demonstrate that our approach achieves the best results on the UCSD Pedestrian2 dataset.

1.4 Dataset

The UCSD Anomaly Detection Dataset was acquired with a fixed camera mounted at a higher location overlooking a sidewalk [15]. The density of pedestrians varies from sparse to very crowded. The UCSD Pedestrian2 dataset contains 16 training and 12 testing video clips, respectively. Each video clip is composed of approximately 150 image frames. Thus, the training dataset consists of a total of 2550 normal frames, which are analyzed by the two neural networks to learn the normal patterns of pedestrians. The testing dataset contains 2010 frames comprised of normal and abnormal frames. These testing frames are fed into the pre-trained model to detect anomalous events. The dataset contains masked images highlighting the abnormal objects suitable for evaluating the performance of different models. All training and testing frames are resized to 256×256. Some examples are shown in Figure 1.1.

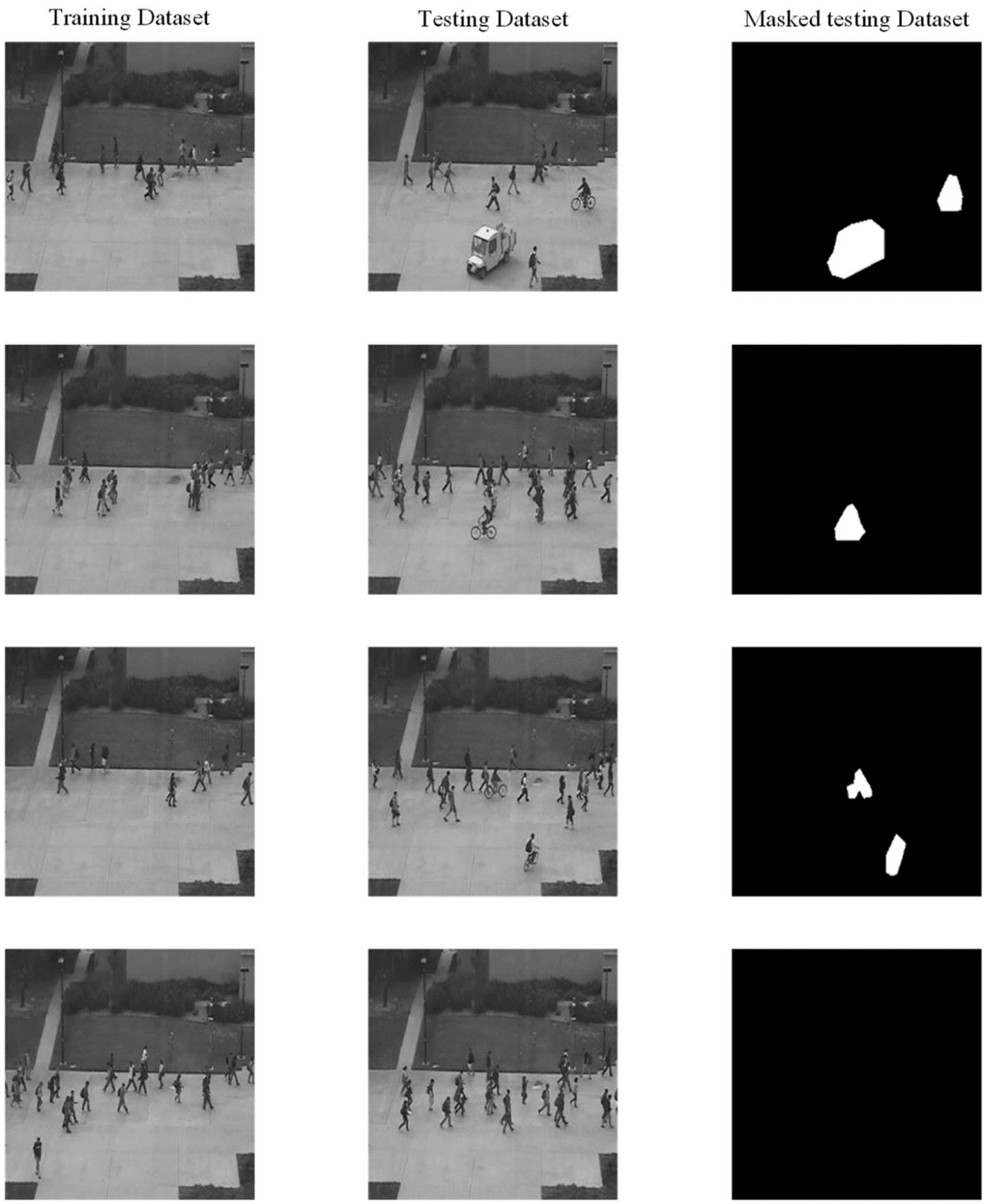


Figure 1.1: Examples of the UCSD Pedestrian2 dataset. Training dataset only contains normal frames, while testing dataset is composed of both normal and abnormal frames. Masked testing dataset highlights all abnormal objects

2 Background

In this chapter, we briefly review convolutional neural networks and generative adversarial networks. Then, we introduce several traditional and deep learning based anomaly detection methods.

2.1 Artificial Neural Networks

Artificial Neural Networks (ANN), also called neural networks, are becoming more and more popular in various applications, such as classification, clustering, object detection and image segmentation.

ANN is inspired by the biological neural system. In most cases, artificial neural networks are self-adaptive systems and they are able to optimize their internal structures based on external information, which is commonly known as learning abilities. In 1943, psychologist McCulloch and mathematical logician Pitts [16] firstly proposed the artificial neural network. It is composed of a large number of neurons connected to each other, each neuron receives multiple input signals including output signals from the other neurons and processes these signals by performing multiplication and addition. Multiple

neurons work together to learn the specific patterns inside the input data. The output of the network performs a task such as classification or regression.

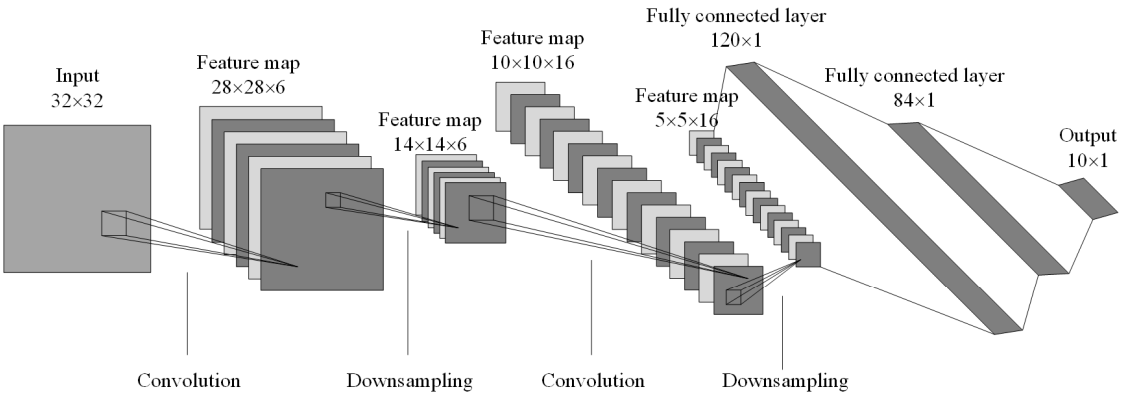
Multi-layer perceptron (MLP) is one of the most famous artificial neural networks. Any MLP includes at least three different types of layers: one input layer, at least one hidden layer and one output layer. A hidden layer is located between the input and output layers, which extracts important features from input signals. The neurons inside each hidden layer transform the input low-level information into high-level features and output the predicted results through a specific activation function. The insertion of hidden layers improves the ability of artificial neural networks and enables a neural network to learn more complex patterns in different tasks. MLP provided a basic artificial neural network architecture for future neural networks. However, the network complexity and the training computation are huge in large networks due to the fully-connected neurons architecture. It is impractical to train a complex and deep artificial neural network when the input data is high dimensional such as images.

2.2 Convolutional Neural Networks

With the development of artificial neural networks and scientists' further research on the human visual system, in 1989, LeCun combined the back-propagation algorithm with the

weight-sharing convolutional neural layer and proposed the convolutional neural network (CNN) [17]. He first applied the convolutional neural network to a zip code recognition system. CNN is a feed-forward neural network and it is comprised of input layers, convolutional layers, pooling layers, activation layers, fully connected layers and output layers. In most cases, multiple convolutional layers and pooling layers are connected alternately, e.g., one pooling layer is added after a convolutional layer and the pooling layer is followed by another convolutional layer. This structure allows convolutional neural networks to effectively analyze and extract critical abstract features from the input images. Moreover, it reduces the size of the data and feeds useful information into the activation and fully connected layers, ensuring the network delivers those high-level image features to the output even for a CNN with many layers and a deep structure. In 1998, based on the Fukushima's research work [18], LeCun trained a CNN using the gradient back-propagation algorithm. This CNN called LeNet-5 is one of the earliest and most classic convolutional neural networks, and it achieved great success on the handwritten digits recognition task [19]. A basic LeNet-5 is shown in Figure 2.1.

In Figure 2.1, convolution is the convolution layer, and Rectified Linear Unit (ReLU) is the activation layer. M and N are constants. Generally, if M and N become larger, the convolutional layers will become deeper, which enables the network to capture more high-level features and improve performance on different tasks.



$$\Big[[[\text{Convolution} \rightarrow \text{ReLU}] * \text{M} \rightarrow \text{AvgPooling}]\Big] * \text{N} \rightarrow \text{Fully Connected} \rightarrow \text{Output}$$

Figure 2.1: The architecture of LeNet-5

Each convolutional layer consists of several different convolutional kernels which are weighing matrices with different sizes, typically 1×1, 3×3 and 5×5. Different convolution kernels are used to extract different latent spatial information from the images. The first several convolution layers aim at capturing low-level local features such as edges or circles, while the following deeper layers are devised to capture high-level features such as hairstyles of humans or eyebrows of cats. The pooling layer provides an approach to reduce the dimension of feature maps and only selects the important data. With the help of convolution and pooling layers, only useful spatial features are retained and fed into the activation layer. The activation layer provides a nonlinear transformation to the neural network, which helps the network learn a non-linear relationship between

inputs and outputs. The current most popular activation functions shown in Figure 2.2 include ReLU, Sigmoid and Tanh functions. ReLU is widely used in many convolutional neural networks.

$$ReLU: y = \begin{cases} x, & x \geq 0 \\ 0 & x < 0 \end{cases} \tag{2.1}$$

$$Sigmoid: y = \frac{1}{1+e^{-x}} \tag{2.2}$$

$$Tanh: y = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.3}$$

Compared with other activation functions, ReLU has several advantages: (1) The derivatives of ReLU are more easily computed, which speeds up the network's training process. (2) Any value less than 0 is set to 0. This enables the ReLU to sparse the neural network and filter out features, which improves the generalization capability and reduces the risk of overfitting. (3) Effectively reduce the probability of the vanishing gradient problem.

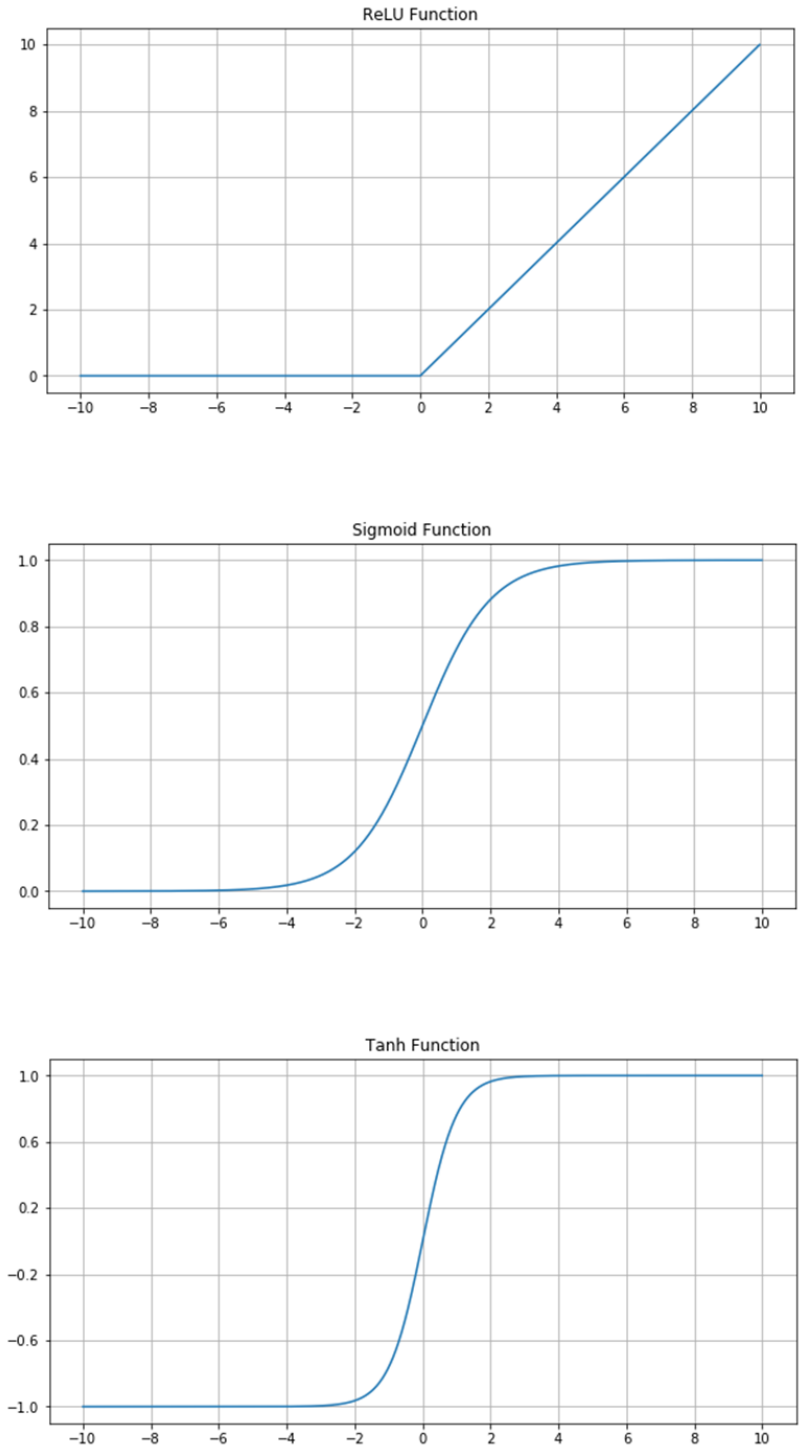
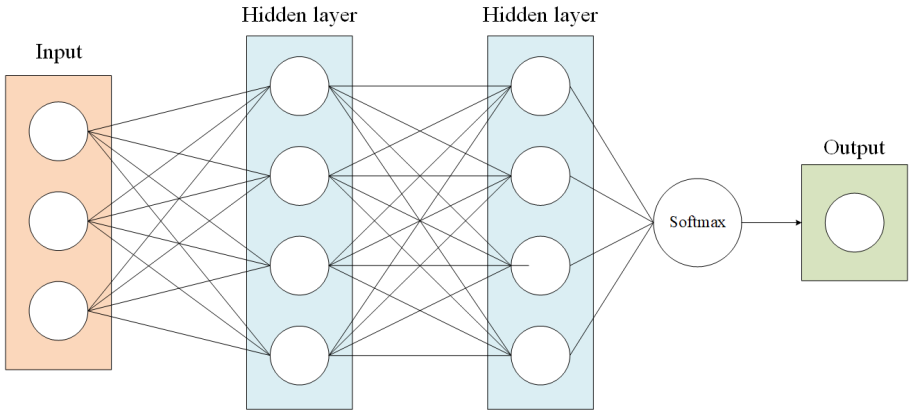


Figure 2.2: ReLU, Sigmoid and Tanh activation functions

Figure 2.3 shows a fully connected neural network implemented by a multilayer perceptron. The high-level image features extracted from the convolutional and pooling layers are converted into a one-dimensional vector as the input to the fully connected neural network. After several nonlinear transformations of hidden layers and the Softmax function, we can obtain the probability of each handwritten digit and use it to complete the classification task.



Input layer→Hidden layer*M→Output layer→Softmax→Output

Figure 2.3: The architecture of the fully connected neural network

In the past decade, convolutional neural networks have been widely used in many image processing areas. AlexNet model [20], GoogLeNet model [21] and VGG model [22] have achieved good results in ImageNet competition, and ResNet model [23] has solved models' degradation problem. A large number of deep learning methods based on

convolutional neural networks have been proposed to solve difficult problems in different areas and they have achieved better results than traditional machine learning algorithms.

2.3 Generative Adversarial Networks

Generative adversarial network (GAN), originally proposed by Ian Goodfellow and his colleagues in 2014 [13], is a two-player generative model where the core idea comes from the zero-sum game. The model architecture of GAN is quite different from that of most CNNs, GAN consists of two neural networks, a generative neural network and a discriminative neural network. The entire network is like a two-player adversarial game, the generator tries to learn the distribution of real data as closely as possible and it converts a one-dimension noise into a generated sample. The discriminator, on the other hand, is a simple binary classifier designed to discriminate whether the input data is from real samples or samples simulated by the generator. In order to win this two-player game, both the generator and the discriminator need to optimize their parameters as much as possible to improve their generative and discriminative abilities, respectively. This learning and optimization process is called finding the Nash Equilibrium between the generator and the discriminator. After several rounds of training, the generator can generate samples almost identical to the real samples, while the discriminator cannot

distinguish the difference between the generated samples and the real samples.

2.3.1 Basic GAN

The main structure of a generative adversarial network is shown in Figure 2.4. The input to the generator is random noise *z*. The inputs to the discriminator are the real samples *x*, and the output of the generator is *G(z)*. Assume that the real data obeys the distribution of *X*. The discriminator outputs 1 if the input of the discriminator is the real sample *x*, and 0 if it is the generated sample *G(z)*.

During the training process, the generator aims to generate fake images *G(z)* that cannot be distinguished from the real images, while the discriminator is used as a supervisor to distinguish which input is the real sample or the generated sample. The generator tries to generate images as real as possible to deceive the discriminator, and the discriminator keeps optimizing itself to distinguish the real images from the fake ones. Based on this design idea, the objective function of GAN is shown as follows.

$$L_{GAN}(G,D) = \min_{G}\max_{D} V(G,D) = E_{x\in X}\left[log\left(D(x)\right)\right] + E_{z\in Z}\left[log\left(1-D\left(G(z)\right)\right)\right], \quad (2.4)$$

where *V(G, D)* is a cross-entropy function of the binary classification. The objective of this loss function is to find a generator that minimizes the Kullback-Leibler Divergence (KL divergence) between the generated distribution and the real distribution, and

discriminator minimizes the cross-entropy loss of the binary classifier. The KL divergence is defined as:

$$KL\left(P\|Q\right)=\sum_{x\in X}P(x)log\left(\frac{P(x)}{Q(x)}\right), \tag{2.5}$$

where *P(x)* and *Q(x)* are two distributions. The KL divergence measures the similarity between two distributions and its value is greater than or equal to 0 where 0 means *P(x)* and *Q(x)* are the same distribution.

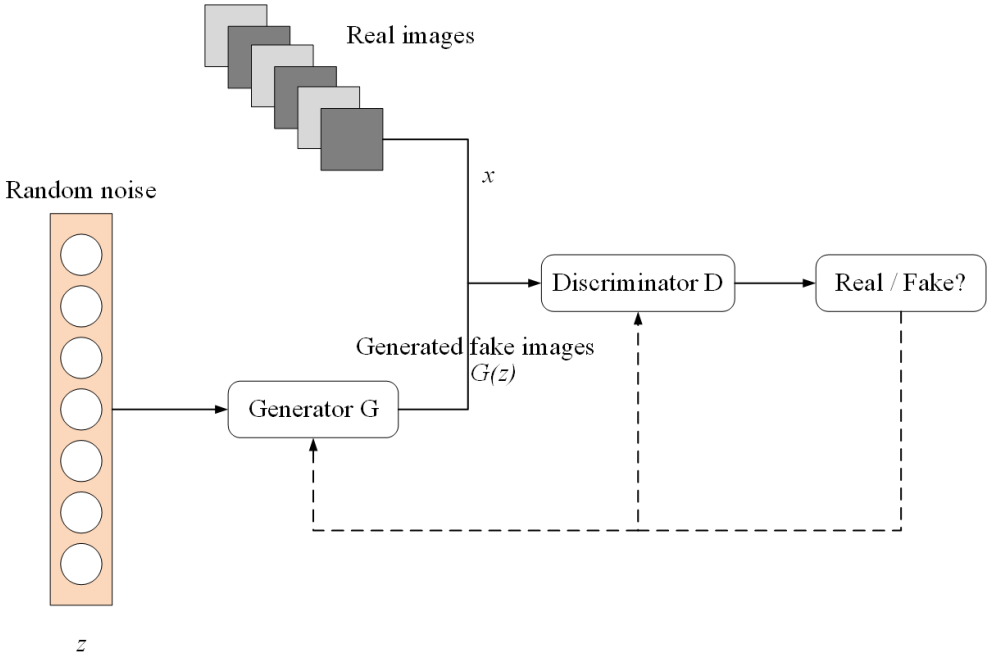


Figure 2.4: The architecture of the generative adversarial network

In the process of optimizing *D*, *D* tries to minimize the output if the input is from the real sample *x* and maximize the output if the input is from the generated sample *G(z)*. On the contrary, *G* has to minimize the loss function *V(G, D)* when the generated data *G(z)* is

fed into *D*. However the discriminator is more prone to overfitting since the generating process of the original GAN is too simple. For improving the performance of GAN and making it robust to more generative tasks, researchers have proposed many improved neural networks based on GAN, such as conditional GAN (CGAN) [24], deep convolutional GAN (DCGAN) [25] and Wasserstein GAN (WGAN) [26], etc.

2.3.2 Conditional Generative Adversarial Network

The original GAN has almost no constraints on the generator. Since the generation process is composed of a series of simple upsampling operations from vectors to images, the generator tends to generate some repetitive but safe samples, which causes the generated images to lack diversity. Moreover, we have no control over the generated results and cannot decide which image to generate. CGAN allows us to control the output of GAN with additional restrictions such as the label of categories. The structure of CGAN is shown in Figure 2.5.

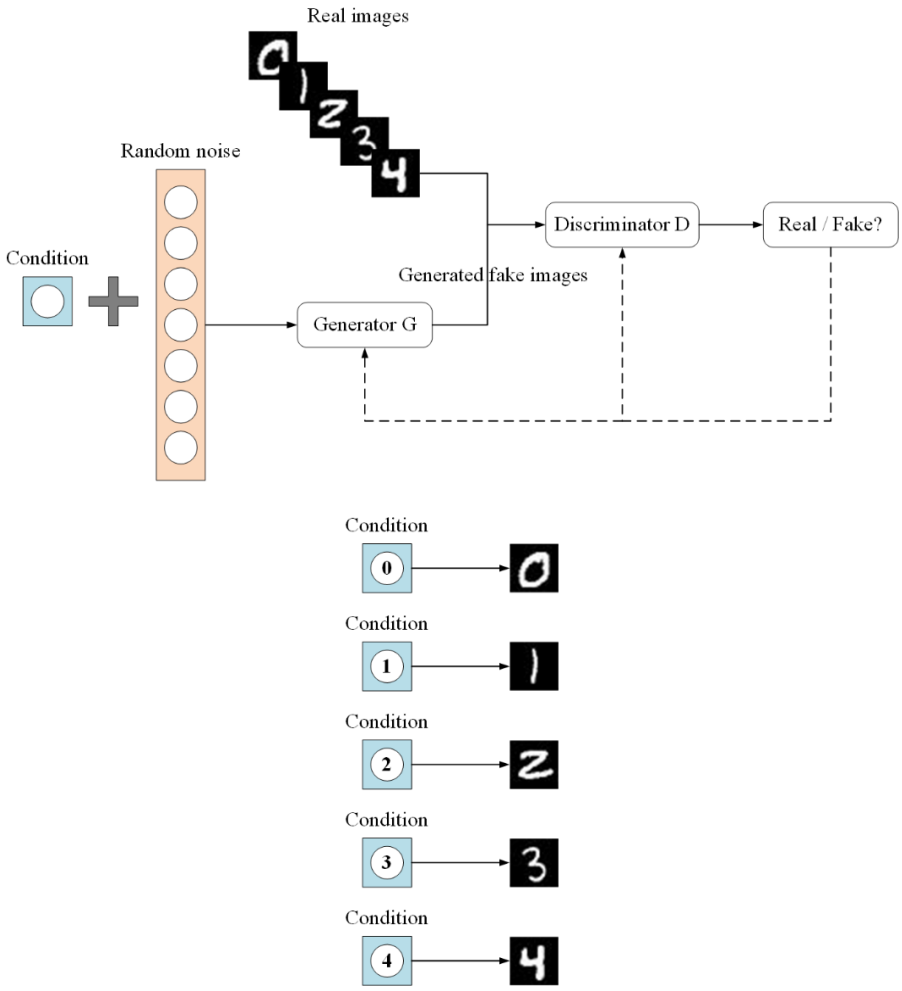


Figure 2.5: The architecture of the conditional generative adversarial network

2.3.3 Pix2pix Network

The input of the previous GANs is random noise, so these networks cannot map images to images. In order to extend GAN to end-to-end tasks and fix this issue, Phillip Isola proposed pix2pix network [27], which is an image-to-image translation network based on

GAN. Pix2pix network uses an encoder-decoder as the network structure of the generator and adds skip connections between the encoder and decoder since pix2pix network needs to map one high-resolution image to another high-resolution image with different spatial features and/or complex color information. Pix2pix network also proposed a new discriminator architecture - which we term a PatchGAN - that only penalizes structure at the scale of patches [27]. It divides each image into N×N patches with equal sizes, then the discriminator distinguishes whether each patch is from the real image or the generated image separately. Finally, the network averages these patches' results as the final output of the discriminator. The objective of pix2pix network is shown as follows:

$$L_{pix2pix}(G,D) = L_{CGAN}(G,D) + \lambda L_{L1}(G), \tag{2.6}$$

where this objective function consists of two parts, the CGAN loss function and the L1 loss function. To ensure the generated images with high resolution and sufficient low-frequency features, pix2pix network adds the L1 loss function to penalize the difference between the generated image and the ground truth.

Pix2pix network cleverly uses the GAN framework to provide a general image-to-image translation framework, which is widely used in semantic segmentation, image reconstruction, anomaly detection and other areas.

2.4 Anomaly Detection Methods

2.4.1 Object Trajectory Methods

Object trajectory detection methods are designed to detect different trajectories of moving objects since the motion patterns of pedestrians are quite different from those of vehicles.

[1] proposes a novel framework to learn semantic scene models by object classification and trajectory clustering. The flowchart of this object trajectory method is shown in Figure 2.6. This approach is composed of two parts, (1) a co-training framework for object detection, and (2) a clustering algorithm based on trajectories' features of different objects. The co-training framework is a two-classifier system, including a Linear Discriminant Analysis (LDA)-based classifier and an Adaptive Boosting (AdaBoost) classifier, which are trained and optimized using independent features.

Firstly, they prepare two groups of unrelated features, namely scene context features and appearance features. Scene context features represent objects' characteristics such as coordinates and velocity. The LDA-based classifier is trained using these features to distinguish different objects. On the other hand, they build a binary classifier based on the Adaboost Algorithm to learn specific representations of each object's appearance

features and to recognize objects. With the help of these two techniques, the co-training algorithm provides a convenient and effective way to build a perfect classifier that is used to make predictions on unlabelled images and to scale up the training dataset.

Secondly, to further cluster each class, they propose a clustering method based on object motion trajectories. Each image is divided into several small patches of the same size. Depending on the parameters and moving patterns of trajectories, different types of trajectories will be grouped into different clusters.

Lastly, after generating each cluster of trajectories, a Mean-shift algorithm is introduced to create a semantic scene model which can draw every main trajectory on each image. Any object with an abnormal trajectory will be detected as an anomalous event.

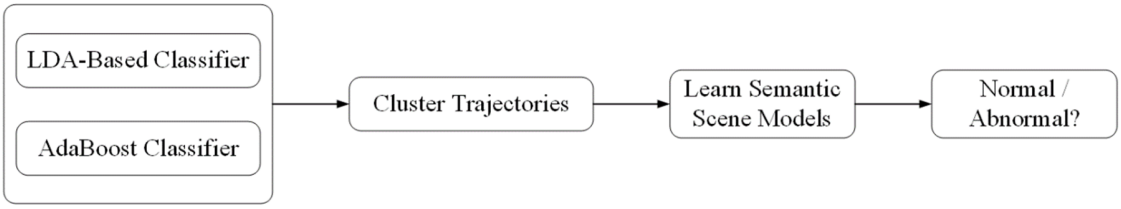


Figure 2.6: The flowchart of the object trajectory method

2.4.2 Handcraft Methods

[4] proposes a classification method for detecting and locating abnormal crowd behaviors

using Social Force model. Social Force model describes crowd behavior as the result of individual interactions, therefore, abnormal crowd behavior is an eccentric state of the crowd interactions [4]. Detecting abnormal social forces will help us find abnormal behaviors and events on each frame.

Normally, tracking the interaction forces of each object is time-consuming and labor-intensive, so they decide to replace each object with a grid of particles that are placed over each image frame. Then they calculate the interaction forces between each particle. During training steps, only normal videos are fed into this method and they collect enough normal crowd behaviors. Finally, they use a bag of words approach to identify which image frame contains abnormal interaction forces.

2.4.3 Deep Learning Methods

In the past 10 years, deep learning algorithms have become more and more popular and successful in many research areas. Scientists and researchers have started to build various CNN-based anomaly event detection methods.

[6] presents a video prediction framework based on GAN to solve anomaly detection tasks. The pipeline is shown in Figure 2.7. This framework includes two components, (1) predicted image reconstruction and (2) optical flow image comparison. The input of the

generator is a short video clip ($I_1, I_2, I_3, \ldots, I_t$) that includes a set of consecutive image frames. For each short video clip, the ground truth is the next frame (I_t), and the target of the generator is to reconstruct an image frame ($\hat{I}_{t+1}$) which is trained to be as similar as possible to the ground truth image. On the other hand, they compute the ground truth optical flow images and the predicted optical flow images, respectively. Since only normal frames are used to train and finetune the generator and discriminator, any abnormal object cannot be well reconstructed in the testing steps. Therefore, this method utilizes the difference between predicted frames and corresponding ground truth frames to detect anomaly events.

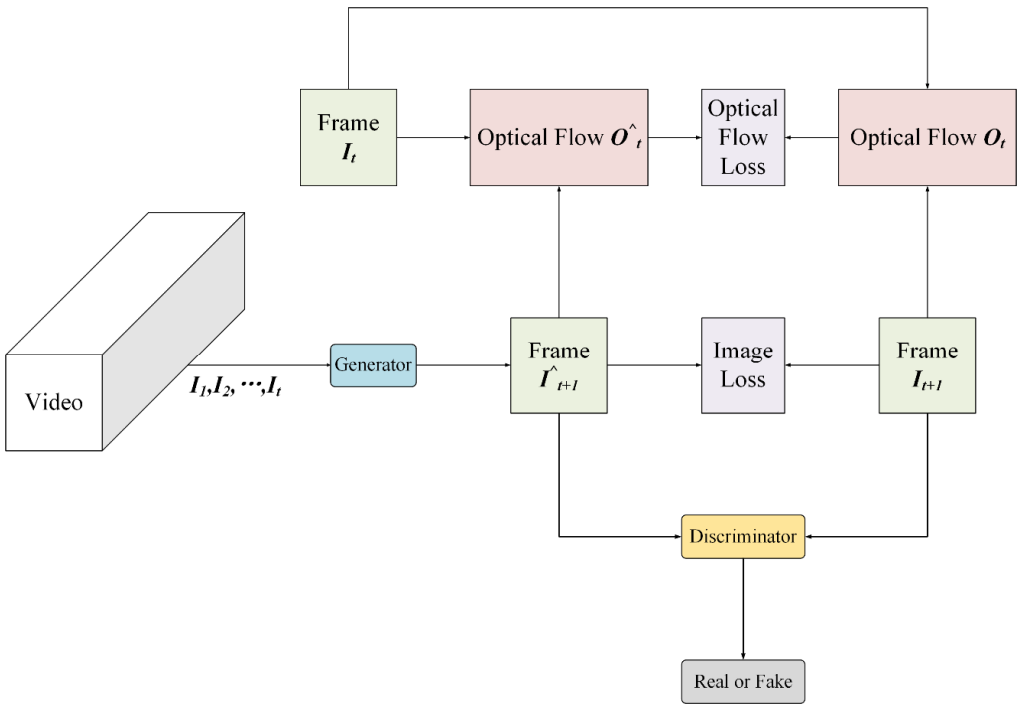


Figure 2.7: The pipeline of the video future prediction network

3 Anomaly Detection on Two-way Multi-input Generative Neural Network

In this chapter, we introduce the architecture of our neural networks, three different loss functions and a patch-based evaluation method. An overview of our neural networks is shown in Figure 3.1. Our model is composed of two neural networks: appearance generation neural network (N^{appe}) and motion generation neural network (N^{motion}). These two neural networks use almost the same network structure, and the biggest difference is the input and label image dataset used in the training steps. In the appearance generation neural network, the network receives a concatenation of two preprocessed frames as input and generates an appearance frame. At the same time, in the motion generation neural network, a motion frame is generated from the input frame which is one of the frames in the original dataset. Then, we detect and locate anomalous events using the generated frames from two neural networks using a new patch-based evaluation method.

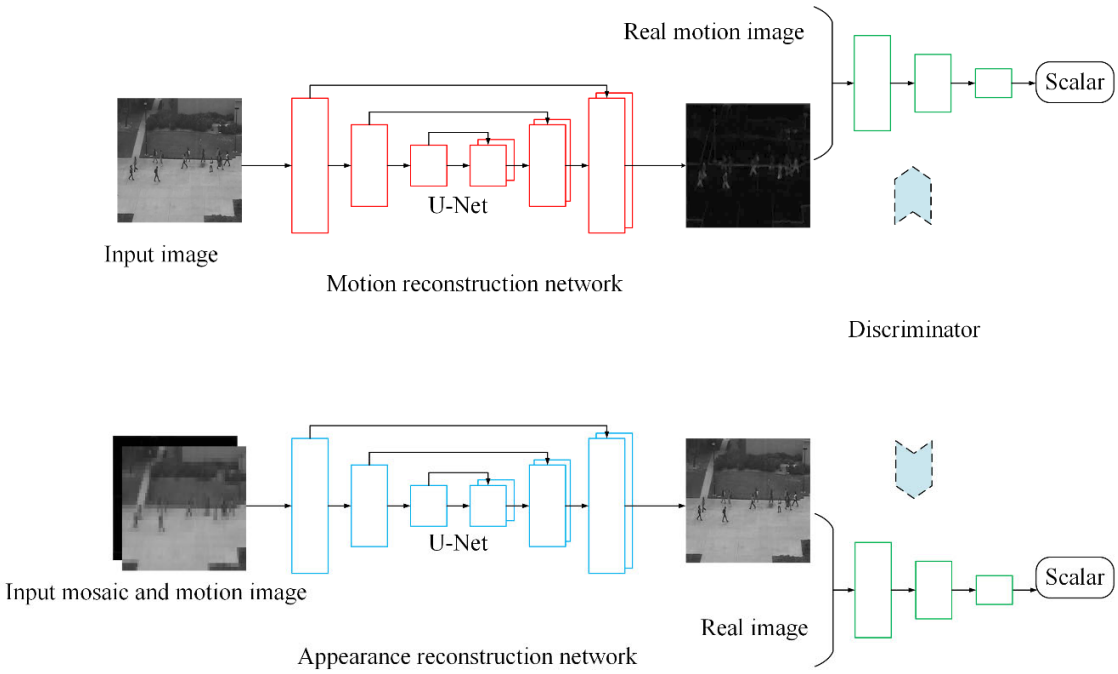


Figure 3.1: An overview of two-way multi-input generative neural network

3.1 Two-way Multi-input Generative Neural Network

3.1.1 Multi-input

Although GAN has proven to be the most successful generative model in many areas, the training process of GAN, especially the generator, is quite hard and prone to overfitting. It is much harder to obtain an acceptable result for a generative model than for a discriminative model. In most cases, the generative model requires more prior knowledge,

such as the distribution of target signals. For example, if we want to build a handwritten digits image classification model, then following Bayes' Theorem,

$$P(B \mid A) = \frac{P(A,B)}{P(A)} = \frac{P(A \mid B)P(A)}{P(A)} \tag{3.1}$$

$$\textit{Discriminative Model Target}: P(B \mid A) \tag{3.2}$$

$$\textit{Generative Model Target}: \frac{P(A,B)}{P(A)} \tag{3.3}$$

As shown in Figure 3.2, the discriminative model is designed to draw a line to distinguish handwritten 0's and 1's. It aims to figure out what's the most optimal solution, e.g. $P(B \mid A)$, to tell the difference between 0's and 1's, while the generative model has to learn the distribution of all 0 and 1 samples throughout the whole data space, e.g. $P(A,B)$.

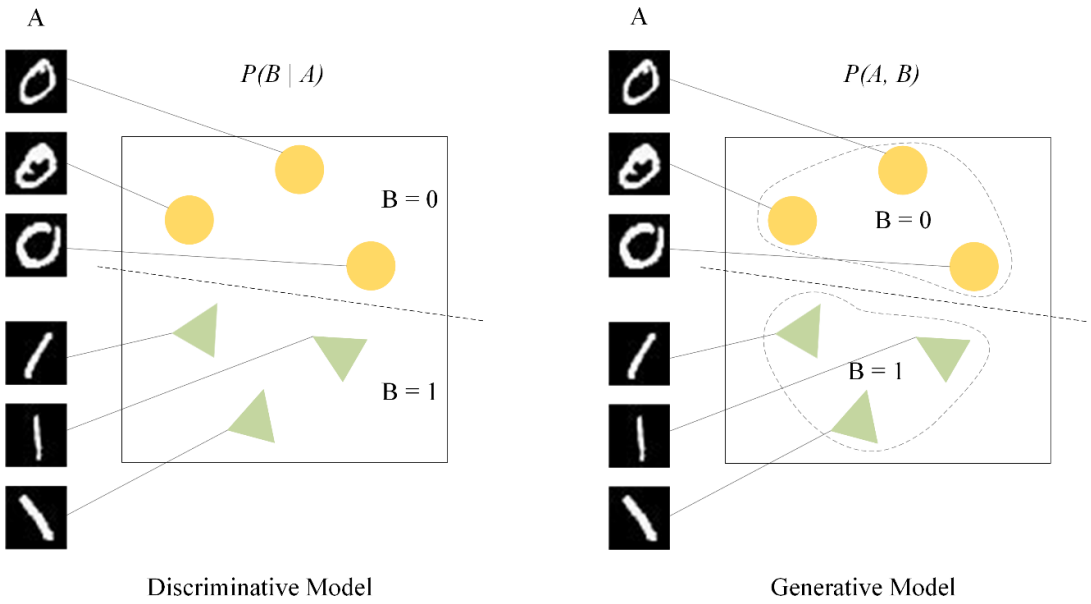


Figure 3.2: The difference between the generative and discriminative model

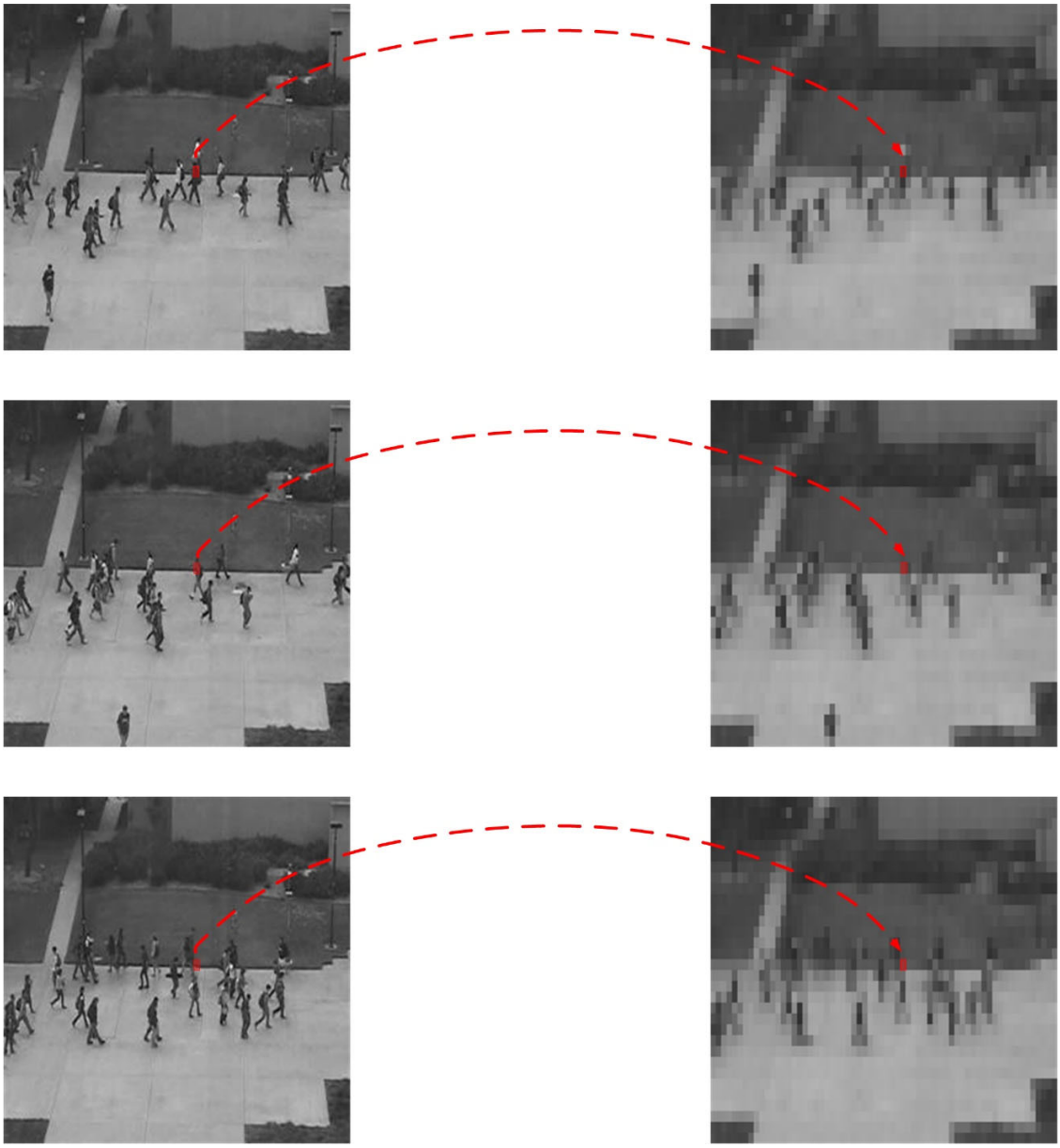


Figure 3.3: Mosaic frames & 8×4 patches

For solving this issue and improving the performance of the generator, the multi-input is added to the appearance generative neural network. The multi-input consists of motion frames and mosaic frames. Mosaic frames are used to provide the grayscale

information of frames to the generator but discard the edge and the shape features of objects, such as humans and cars, in the pictures. Simultaneously, motion frames aim at keeping the motion details and shape features of any object while removing all object-related grayscale data.

As shown in Figure 3.3, all frames are divided into multiple patches with the size of 8×4 and we compute the average pixel intensity of each patch. Then all pixels in each patch are assigned this mean value so that every mosaic frame looks like one low-resolution image. The generator is only able to receive the grayscale level information from mosaic inputs.

The process of forming motion frames is a bit more complex than the process of generating mosaic frames. Figure 3.4 illustrates the details of the process.

$$ConvF_t = F_t \otimes K_{24} \tag{3.4}$$

$$R_t = mean\left(abs\left(ConvF_t - F_{t+1}\right)\right) \tag{3.5}$$

$$Mt_t = MedianFilter\left(Close\left(Open\left(R_t\right)\right)\right) \tag{3.6}$$

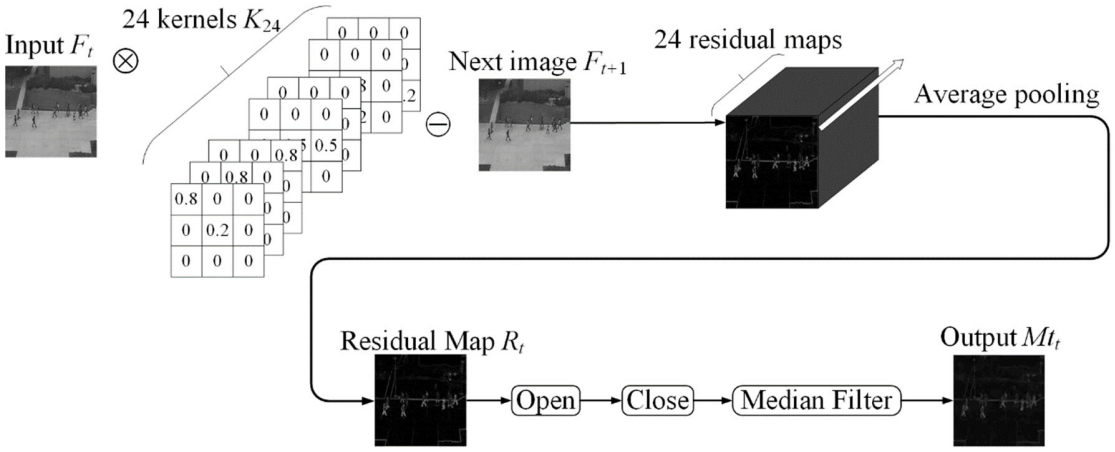


Figure 3.4: The process of constructing motion frames

Inspired by [28], a fixed kernel set (K_{24}) comprised of 24 kernels is pre-constructed and all kernels are concatenated with each other. Each original image (F_t) is convolved with these 24 kernels and extracts 24 feature maps. The feature maps are subtracted from the next frame (F_{t+1}) separately to obtain 24 residual maps. Then we compute the absolute pixel intensity of each residual map and average these residual maps to get one averaged residual map (R_t). Finally, for generating images (Mt_t) with smooth motion, we apply the open and close algorithm and 5×5 median filter on the averaged residual map.

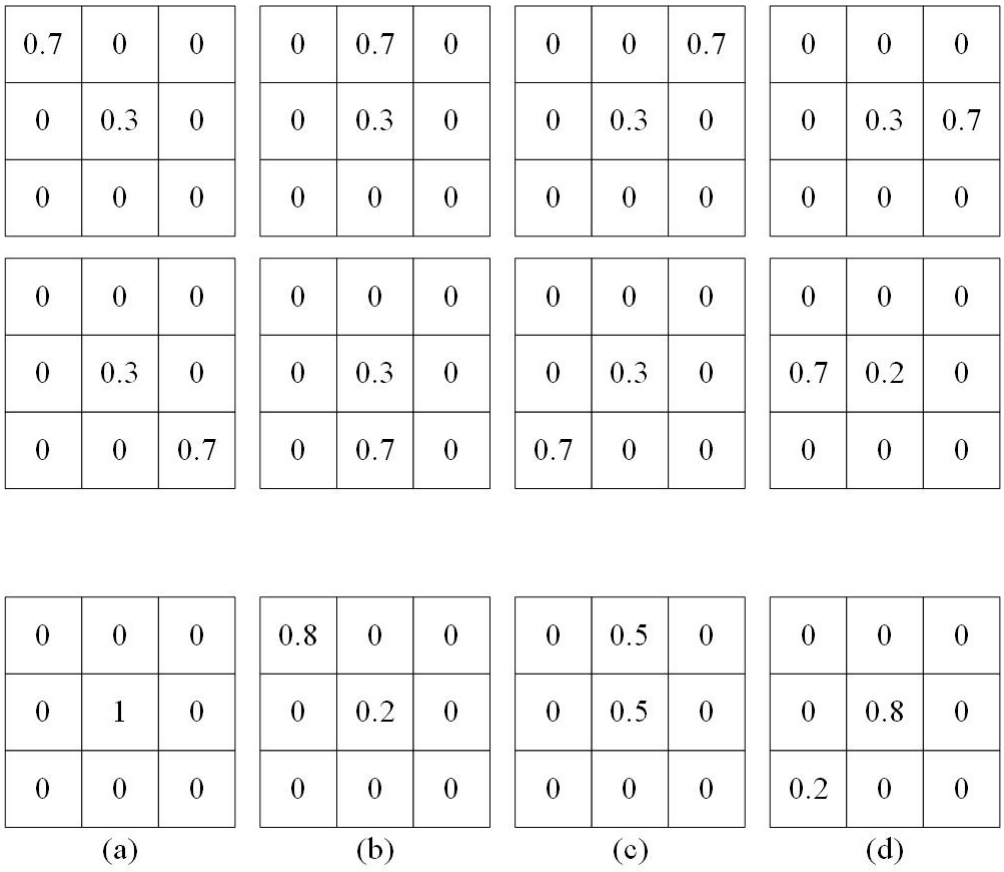


Figure 3.5: Examples of pre-constructed kernel set

Convolutions with a fixed kernel set are designed to simulate small pixels movement following [28]. Some kernels are shown in Figure 3.5. Kernel (a) corresponds to no moving, it is used to capture the static features, kernel (b) means the object moves 0.8 step to the figure's lower right corner, kernel (c) means the object moves 0.5 step to the figure's bottom, and kernel (d) means the object moves 0.2 step to the figure's upper right corner, respectively. With the help of convolutional kernels and the image processing techniques, the generated motion frames are able to detect any static features and all

possible motion steps and directions.

3.1.2 Generators

The generator is used to build well reconstructions of normal objects and poor reconstructions of abnormal objects. Before training, all input images are normalized to -1 and 1. G^{appe} is the multi-input generator in the appearance generation network, while G^{motion} is the generator in the motion generation network. Figure 3.6 shows the overall structure of the two generators.

The input of G^{appe} is the multi-input composed of mosaic frames and motion frames. Mosaic frames are designed to represent the grayscale level of different objects, such as humans and bikers, while motion frames are mainly focusing on capturing the shape and motion features of objects. Compared to single optical flow images, the multi-input ensures the generator receives enough information to reconstruct frames. G^{motion} takes an original frame as input and outputs a motion generation. UNet is set to be the architecture of these two generators, and the details of the generator are shown in Figure 3.7.

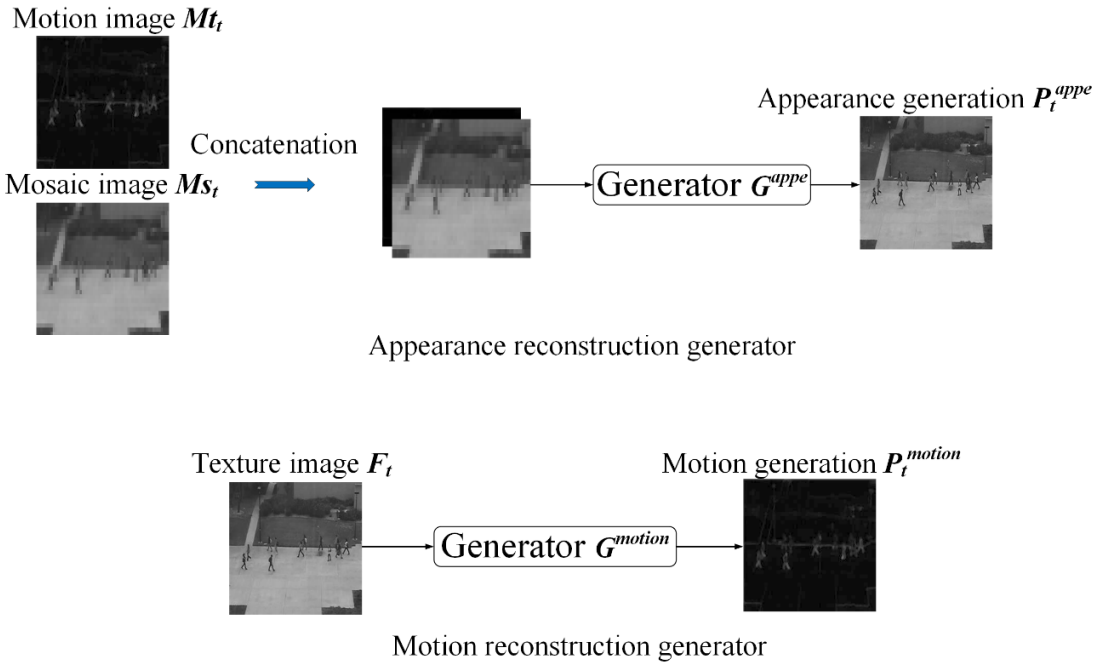


Figure 3.6: The overall structure of the two generators

Specifically, Ms_t and Mt_t are the *t*-th frame of the mosaic and motion images, respectively, and F_t is the *t*-th frame of the original images. Mt_t is generated through the fixed kernel set and several image processing algorithms.

In the appearance generation neural network, the multi-input I_t^{appe} is the concatenation of Ms_t and Mt_t. The output is P_t^{appe}.

$$I_t^{appe} = Ms_t \oplus Mt_t \tag{3.7}$$

$$P_t^{appe} = G^{appe}\left(I_t^{appe}, z\right) \tag{3.8}$$

where $\oplus$ means concatenation operation and *z* is random noise.

In the motion generation neural network, the original image (F_t) is fed into the

generator and the output is P_t^{motion}.

$$P_t^{motion} = G^{motion}\left(F_t, z\right) \tag{3.9}$$

As shown in Figure 3.7, the generator has an encoder-decoder structure and skip connections are designed to connect encoder layers with the corresponding decoder layers which have the same-sized feature maps. The encoder is a convolutional neural network with several similar modules, each module consists of convolutional layers, activation layers and batch normalization layers. Following [27], the batch normalization layer is not added to the first module. Instead of using pooling layers to downsample the feature maps' sizes, we use the strided convolutional layers, since pooling layers will remove some important spatial or shape features, which is quite harmful to the decoder's image reconstruction. The activation function is Leaky ReLU shown in Figure 3.8.

$$Leaky\ ReLU: y = \begin{cases} x, & x \geq 0 \\ ax, & x < 0, a \in [0,1] \end{cases} \tag{3.10}$$

In our network, a is set to 0.2. Moreover, Leaky ReLU is used to avoid weights becoming zero since ReLU activation function will sparse the weights and filter out useful weights.

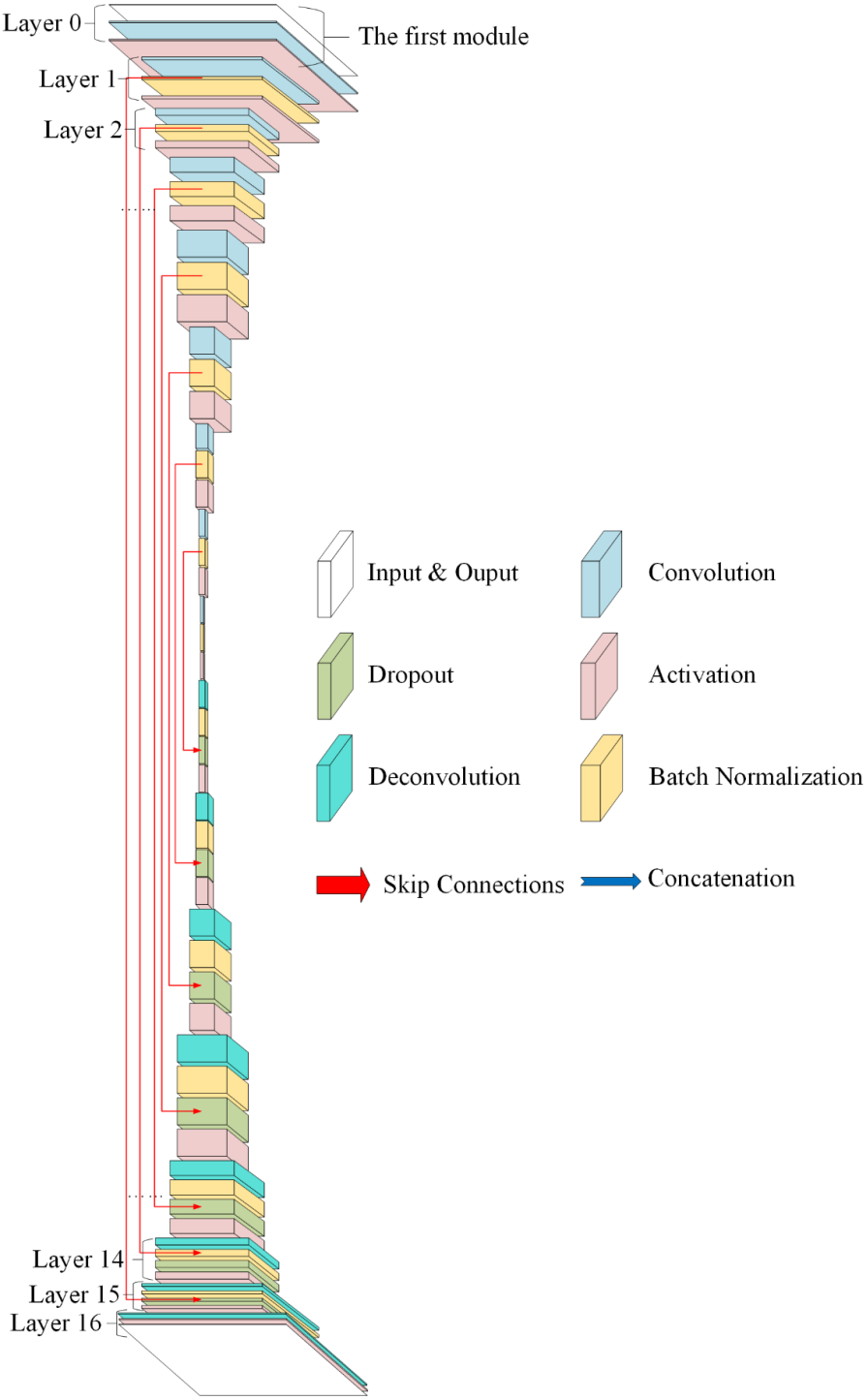


Figure 3.7: The details of the generator architecture

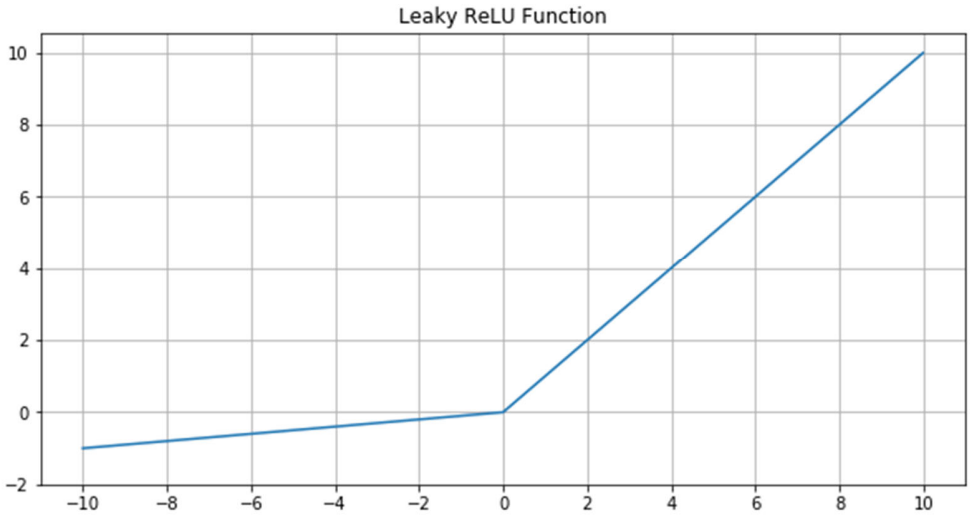


Figure 3.8: Leaky ReLU activation function

As for the decoder, it is composed of deconvolutional layers, activation layers, dropout layers and batch normalization layers. Dropout layers not only reduce the risk of overfitting but also produce some random noises in the generator. ReLU is set as the activation function and it is used to accelerate the speed of convergence and quickly filter out features.

$$ReLU: y = \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases} \tag{3.11}$$

Skip connections are one of the most important components in our network. With the help of skip connections, local information and some lower-level features are passed from the downsampling layers to the upsampling layers, which helps the decoder layers and improves the quality of generated images. Specifically, layer l_i is concatenated with layer

l_{n-i}, where n is the total number of modules in the generator and *i* is the *i*-th module.

3.1.3 Discriminators

Let D^{appe} and D^{motion} be the discriminator of N^{appe} and N^{motion}, these two discriminators have the same structure shown in Figure 3.9. We use a basic binary classifier to distinguish generated images from realistic images. This binary classifier is composed of several convolutional layers and fully connected layers, the output is a scalar between 0 and 1 representing the input image as normal or abnormal.

D^{appe} or D^{motion} takes realistic images or generated images as input. An output larger than 0.5 means the input is a "real" image with a higher probability, if not, the input is recognized as a "fake" image produced by the generator.

$$S_t^{appe} = D^{appe}\left(P_t^{appe}, F_t\right)\begin{cases} > 0.5, & normal \\ \leq 0.5, & abnormal \end{cases} \tag{3.12}$$

$$S_t^{motion} = D^{motion}\left(P_t^{motion}, Mt_t\right)\begin{cases} > 0.5, & normal \\ \leq 0.5, & abnormal \end{cases} \tag{3.13}$$

P_t^{appe} and P_t^{motion} are the output of G^{appe} and G^{motion} respectively. S_t^{appe} and S_t^{motion} are the output of D^{appe} and D^{motion} respectively.

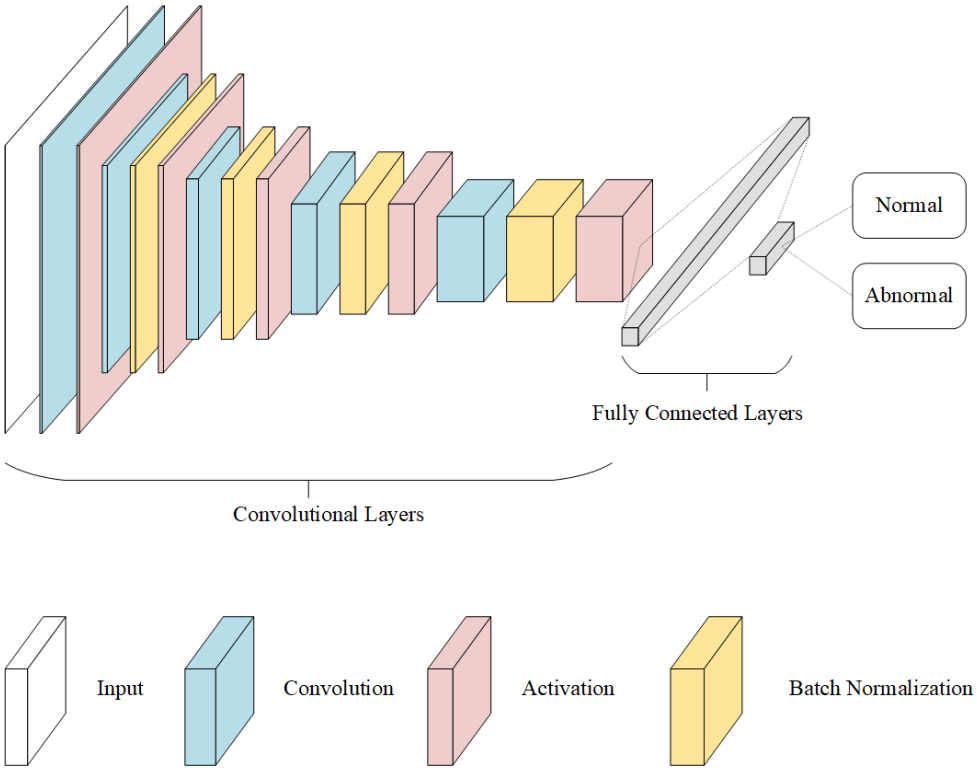


Figure 3.9: The details of the discriminator architecture

3.2 Loss Function

The loss function is composed of three parts, the objective of a conditional GAN, L1 loss function and Structural Similarity Index Measure (SSIM) loss function.

The objective of a conditional GAN is expressed as follows:

$$\begin{aligned} L_{CGAN}(G,D) &= \min_{G}\max_{D} V(G,D) \\ &= E_{x\in X}\left[log\left(D(x\mid y)\right)\right] + E_{z\in Z}\left[log\left(1-D\left(G(z\mid y)\right)\right)\right], \end{aligned} \tag{3.14}$$

where X is the dataset of realistic images and Z is the dataset of generated images (also

called fake images). This function could be divided into two parts: (1) the optimization of the discriminator is achieved by $\max_D V(G,D)$, $V(G,D)$ is the objective function of D. The first term ($E_{x\in X}\left[log\left(D(x\mid y)\right)\right]$) represents the expectation of the probability which the real images are labelled as "True" by D, so this term should be as close to 1 as possible. And the second term ($E_{z\in Z}\left[log\left(1-D\left(G(z\mid y)\right)\right)\right]$) means D should classify all fake images generated by G as "False" and $D\left(G(z)\right)$ should be as close to 0 as possible. (2) the generator is optimized through $\min_G\left(\max_D V(G,D)\right)$, which is $\min_G\left(optimizedD\right)$. G tries to generate more realistic images to fool D.

Some previous research results have found that the mix of the GAN objective and some traditional loss functions is beneficial to generate more realistic images. Pix2pix network verifies L1 distance encourages less blurring than L2 distance on reconstruction [27].

$$L_{L1}(G)=\|I-J\|, \tag{3.15}$$

where I and J are the generated image and real image respectively.

At the same time, L1 loss only focuses on reconstructing less blurry figures, one image still has complex luminance, contract and structure features. Therefore, a new loss function based on SSIM is added to improve the performance of our generators.

The SSIM index was firstly proposed by Zhou Wang et. in 2004 [29]. It is composed of three main components: luminance (l), contrast (c) and structure (s).

$$SSIM(I,J) = l(I,J)^{\alpha} \cdot c(I,J)^{\beta} \cdot s(I,J)^{\gamma}, \tag{3.16}$$

$$l(I,J) = \frac{2\mu_I\mu_J + c_1}{\mu_I^2 + \mu_J^2 + c_1}, \tag{3.17}$$

$$c(I,J) = \frac{2\sigma_I\sigma_J + c_2}{\sigma_I^2 + \sigma_J^2 + c_2}, \tag{3.18}$$

$$s(I,J) = \frac{\sigma_{I,J} + c_3}{\sigma_I\sigma_J + c_3}, \tag{3.19}$$

where c_1, c_2 and c_3 are constant. We set the weights α, β, γ to 1, then the SSIM index will become:

$$SSIM(I,J) = \frac{(2\mu_I\mu_J + c_1)(2\sigma_{I,J} + c_2)}{(\mu_I^2 + \mu_J^2 + c_1)(\sigma_I^2 + \sigma_J^2 + c_2)}, \tag{3.20}$$

where μ_I and μ_J is the mean value of *I* and *J* respectively, σ_I and σ_J are the variance of *I* and *J* respectively, $\sigma_{I,J}$ is the covariance of *I* and *J*. The value of the SSIM index is between 0 and 1 where 1 means *I* and *J* are the same, while 0 means they are completely different. We hope our SSIM loss function is as close to 0 as possible when the discriminator cannot distinguish generated frames from realistic images, and vice versa. We define the SSIM loss function:

$$L_{SSIM}(G) = 1 - SSIM(I,J) = 1 - \frac{(2\mu_I\mu_J + c_1)(2\sigma_{IJ} + c_2)}{(\mu_I^2 + \mu_J^2 + c_1)(\sigma_I^2 + \sigma_J^2 + c_2)} \tag{3.21}$$

Our final loss function is shown as follows:

$$L_{TMGNN}(G,D) = \lambda_1 L_{CGAN}(G,D) + \lambda_2 L_{L1}(G) + \lambda_3 L_{SSIM}(G), \tag{3.22}$$

where λ_1, λ_2 and λ_3 are constant, and we assign 1, 10 and 20 to these parameters respectively.

3.3 Anomaly Detection Evaluation Method

In this section, we will introduce the pixel-based and image-based evaluation methods and propose a new patch-based anomaly events detection and localization method.

3.3.1 Pixel-based and Image-based Evaluation Methods

The pixel-based and image-based evaluation methods are frequently used in anomaly events detection and localization tasks. Most evaluation methods could be divided into these two categories.

To identify which frame contains abnormal objects, the pixel-based evaluation method is designed to compare the pixel-by-pixel difference between reconstructed images and corresponding ground truth images. Let Re_t and Or_t be the t-th frame of reconstruction and ground truth respectively. As shown in (3.23), this residual image ($Diff_t$) is merely defined by subtracting the Re_t and Or_t. To ensure this method is able to capture all abnormal reconstructed pixels, an absolute value operation is added to remove

any negative sign of pixels inside $Diff_t$.

$$Diff_t = \left| Re_t - Or_t \right| \tag{3.23}$$

However, each residual image has a different maximum value and it's not fair to directly compare their results, so any residual image should be normalized firstly into the range [0, 1] as follows:

$$normDiff_t = \frac{1}{\max_t} Diff_t, \tag{3.24}$$

where $\max_t$ is the maximum pixel value of the *t*-th frame.

In most GANs, the generator cannot reconstruct the images perfectly and the generated images still have some errors inside. Therefore, two thresholds are introduced in this pixel-based evaluation method, one is the pixel intensity threshold ($Thres_{val}$) and another one is the total number threshold ($Thres_{num}$). If a pixel intensity is higher than or equal to $Thres_{val}$, this pixel will be labelled as abnormal. In each residual image, the method counts the total number of abnormal pixels. Any image with a total number larger than or equal to $Thres_{num}$ will be detected as an abnormal image.

$$abPixel = pixelVal \geq Thres_{val} \tag{3.25}$$

$$abImage = total\ number\ of\ abPixel \geq Thres_{num} \tag{3.26}$$

Compared with the pixel-based evaluation method, the image-based method is easier to compute. It usually computes a regularity score with a threshold to distinguish which

frame is abnormal. The reconstruction error (E_t) of *t*-th frame of one video sequence is defined by the Euclidean distance between Re_t and Or_t .

$$E_t = \left\| Re_t - Or_t \right\|_2 \tag{3.27}$$

Then the maximum and minimum reconstruction errors of this video sequence are derived. All reconstruction errors are normalized between 0 and 1and the regularity score (S_t) is shown as follows:

$$S_t = \frac{E_t - \min_t E_t}{\max_t E_t - \min_t E_t} \tag{3.28}$$

The regularity score is used to detect if this input frame contains any anomalous objects. If S_t is higher than or equal to the threshold, then the input frame is labelled as abnormal, otherwise, it is a normal frame.

3.3.2 Patch-based Evaluation Method

We propose a new patch-based anomaly events detection and localization method which can better capture abnormal frames and localize anomalous objects.

There are two main benefits to our method: (1) Our method utilizes a patch to replace a pixel. Since abnormal objects usually contain tens of hundreds of pixels, it is not reliable to judge the input frame with one abnormal pixel as an abnormal frame. Instead

of using pixels, we decide to use patches to detect whether there is any abnormal object inside each patch. (2) Our method is able to capture abnormal frames and localize the anomalies at the same time. The image-based evaluation method could only catch abnormal frames, but it cannot indicate the position of abnormal objects.

During the testing steps, we only use the two generators (G^{appe}, G^{motion}). Firstly, the residual frames are defined by computing the absolute difference between each ground truth frame and corresponding generated frame. The two residual frames ($Diff_t^{appe}$, $Diff_t^{motion}$) are shown as follows:

$$Diff_t^{appe} = \left| P_t^{appe} - F_t \right|, \tag{3.29}$$

$$Diff_t^{motion} = \left| P_t^{motion} - Mt_t \right|, \tag{3.30}$$

where P_t^{appe} and P_t^{motion} are the generated output of G^{appe} and G^{motion} respectively, F_t and Mt_t are the original frame and motion frame respectively.

Secondly, all residual frames should be scaled to the range of [0, 1]. The training and testing dataset is composed of several video clips and each video clip (V) contains hundreds of frames. For each video clip, we find its maximum pixel value and every frame in his video clip should be divided by this maximum value to map all pixel intensity to the range of [0, 1].

$$m_v^{appe} = \max_V Diff_t^{appe} \tag{3.31}$$

$$m_v^{motion} = \max_V Diff_t^{motion} \tag{3.32}$$

$$normDiff_t^{appe} = \frac{1}{m_v^{appe}} Diff_t^{appe} \tag{3.33}$$

$$normDiff_t^{motion} = \frac{1}{m_v^{motion}} Diff_t^{motion} \tag{3.34}$$

Since there are two networks N^{appe} and N^{motion} in our method, we add them together and assign different weights to the two normalized residual frames $normDiff_t^{appe}$ and $normDiff_t^{motion}$.

$$normDiff_t = \eta_1 normDiff_t^{appe} + \eta_2 normDiff_t^{motion}, \tag{3.35}$$

where η_1 and η_2 are the two weights to control the ratio between the two normalized residual frames, they are set to 0.2 and 0.8 respectively.

Thirdly, we prepare a patch with the size of 32×16 and move this patch to quickly scan each normalized residual frame $normDiff_t$ column by column and row by row like a sliding window shown in Figure 3.10. Then we compute the average pixel intensity of all patches in each normalized residual frame. Any patch with an average value higher than the pixel intensity threshold ($Thres_{val}$) will be recorded as an abnormal patch. After the sliding window goes through the entire frame, we count the sum (N_{ab}) of the number of abnormal patches, if N_{ab} is higher than or equal to the total number threshold ($Thres_{num}$), this frame will be considered an abnormal image.

$$\begin{cases} N_{ab} \geq Thres_{num}, & abnormal \\ N_{ab} < Thres_{num}, & normal \end{cases} \tag{3.36}$$

One patch is too small to cover an abnormal object and it is not enough to distinguish whether a frame is normal or abnormal. We set $Thres_{num}$ to 210 to make sure our abnormal patches could highlight all abnormal events.

By using this patch-based evaluation method, any anomaly event in the frame is detected and localized.

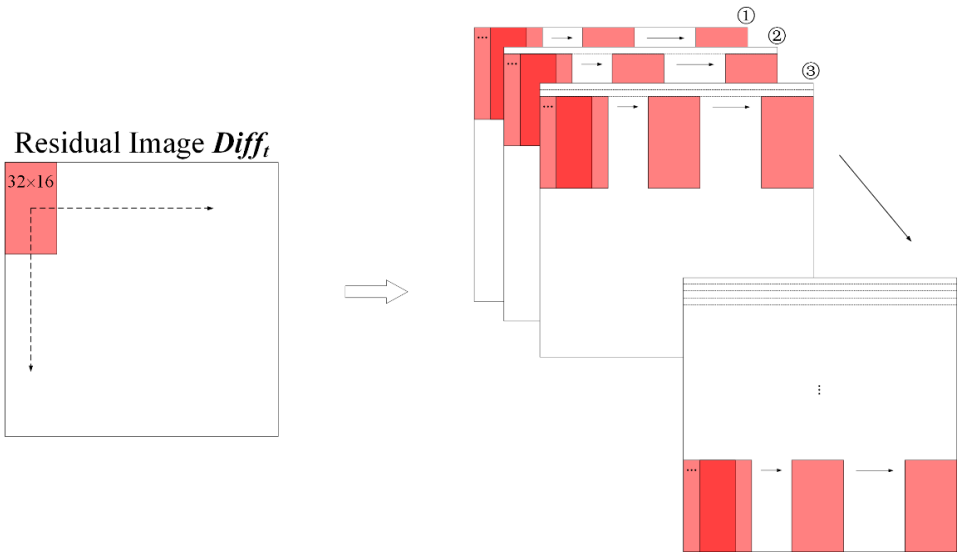


Figure 3.10: Patch-based evaluation method

4 Experimental Results

In this chapter, we introduce a few evaluation metrics and the experimental platform and parameters which we use to train our model. Then we compare the model performance on different evaluation metrics with previous papers' results.

4.1 Evaluation Metrics

For assessing the performance of classification models, researchers typically utilize the confusion matrix which allows visualizing the accuracy of the models.

		Predicted Condition	
Actual Condition	P+N = PP+PN	Abnormal (PP)	Normal (PN)
	Abnormal (P)	True Positive (TP)	False Negative (FN)
	Normal (N)	False Positive (FP)	True Negative (TN)

Figure 4.1: Confusion matrix

The confusion matrix is shown in Figure 4.1, it is a table with 4 different combinations of the actual and predicted cases. In most classification tasks, the positive category is the abnormal class, such as the probability of a future earthquake.

Some widely used evaluation metrics are shown as follows:

$$Accuracy = \frac{TP + TN}{P + N} \tag{3.37}$$

$$Precision = \frac{TP}{TP + FP} \tag{3.38}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.39}$$

$$F1\ Score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{3.40}$$

Accuracy is the most common metric used to explicitly represent the performance of a classification model. However, when we have an extremely imbalanced dataset in which actual positive cases are rare, for example, only occupying less than 1% of the total data, using accuracy will cause a big problem in training the models. As shown in Figure 4.2, there are 99 instances of Class "N" and only 1 instance of Class "P". Even if the model classifies all cases to Class "N", we will still get 99% accuracy. This will mislead the model to ignore the patterns of Class "P".

	Predicted Condition		
Actual Condition	P+N = PP+PN	Abnormal (PP)	Normal (PN)
	Abnormal (P)	0	1
	Normal (N)	0	99

Figure 4.2: An example of a confusion matrix

F1 Score is proposed to solve this issue. It provides robust evaluation results for imbalanced datasets since it not only considers the precision but also the recall. Following

(4.4), if the model classifies all instances to Class “N”, then we will get a 0% F1 Score which punishes the model to pay more attention to the skewed class, such as Class “P”. To obtain the largest F1 Score, we need to set a threshold as a probability boundary between normal and abnormal classes, any input with an output probability lower than this threshold will be detected as abnormal and vice versa.

The Receiver Operating Characteristic Curve (ROC), Area Under the Receiver Operating Characteristic Curve (AUC) and Equal Error Rate (ERR) are three additional common metrics to evaluate the ability of classification models. As shown in Figure 4.3, AUC is calculated as the area underneath ROC which shows the relationship between the True Positive Rate (TPR) and False Positive Rate (FPR). A large AUC means the model achieves higher accuracy. To plot the ROC, a set of TPR and FPR is computed by changing the threshold values. Different thresholds correspond to different TPR and FPR values. For each threshold, the TPR and FPR are plotted on the y-axis and x-axis respectively. AUC and ERR do not require finding a specific threshold unlike the F1 Score.

$$TPR = \frac{TP}{P} = \frac{TP}{TP + FN} \tag{3.41}$$

$$FPR = \frac{FP}{N} = \frac{FP}{FP + TN} \tag{3.42}$$

4.2 Experimental Platform

All of the training and experiments are completed using an Intel Core i7-8700 CPU and an Nvidia GTX 1070 GPU on the Windows 10 platform. All neural networks are written in Python using TensorFlow 2.0, Keras, Numpy and OpenCV library.

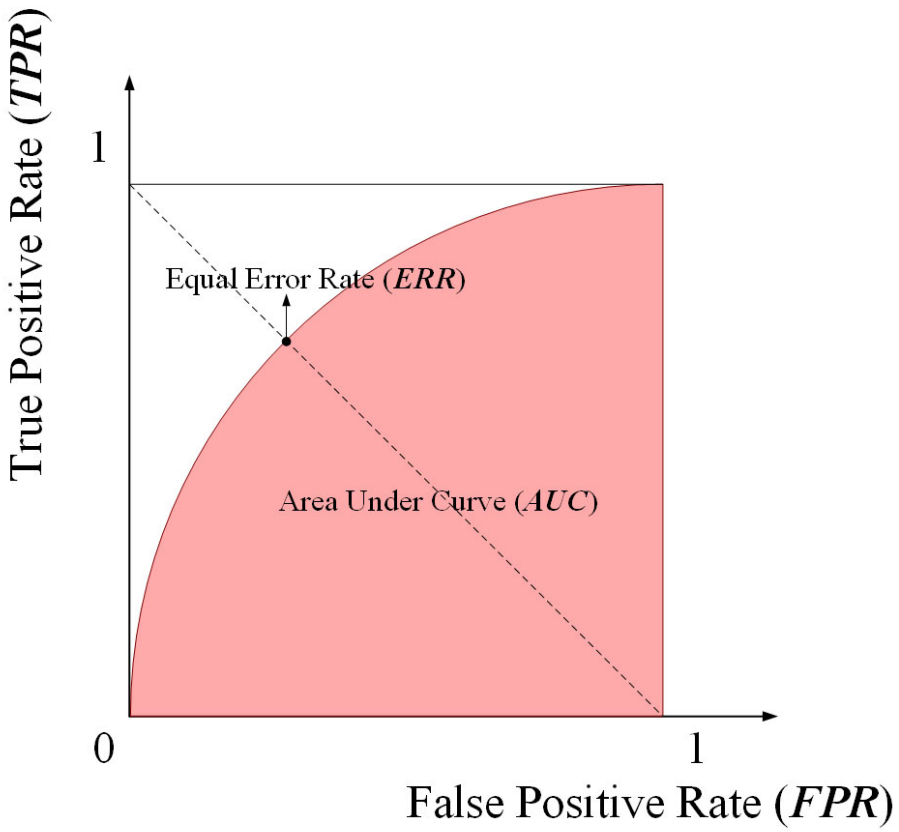


Figure 4.3: An example of ROC, AUC and ERR

4.3 Parameter Selection

All training and testing datasets are grayscale frames and resized to 256×256 pixels. The training process is performed on Adam Optimizer with a learning rate of 0.0001 and 0.0002 in the generator and discriminator, respectively. The batch size is set to 16. In the testing steps, only generators are used for reconstruction. We choose AUC and EER to evaluate the anomaly events detection and localization performance of our model.

4.4 Experimental Results

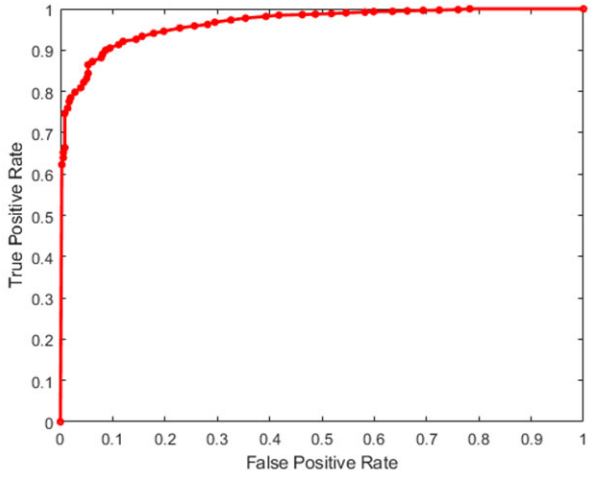
For the experiments, we use our new patch-based anomaly events detection and localization method to test the ability of TMGNN on the UCSD Ped2 dataset. We evaluate our model in two criteria: (1) Frame-level criterion. If the number of abnormal patches in one frame is larger than or equal to 210, then this frame will be classified as abnormal. (2) Pixel-level criterion. Achieving a higher score on this criterion is harder than that on the first one. A frame is a true positive frame, only if it satisfies two conditions. The first one is it should be classified as an abnormal image. The second one is the evaluation method must identify at least 40% area of anomalous objects.

Our training dataset contains 16 video clips and the testing dataset is composed of 12 video clips. Each video clip is comprised of around 150 frames. Since the training dataset only has normal frames, a total of 2550 normal frames are fed into the network. In the testing steps, all testing frames are input into the trained generator and we obtain 2010 generated frames. Then following the detailed process of Section 3.3.2, we utilize the patch-based evaluation method on the normalized residual frames to test the performance of our trained model. By changing the value of $Thres_{val}$, we are able to draw a Receiver Operation Characteristic Curve and compute its corresponding Area Under Curve and Equal Error Rate. A lower value in EER and a higher value in AUC represent a better performance on a classification model.

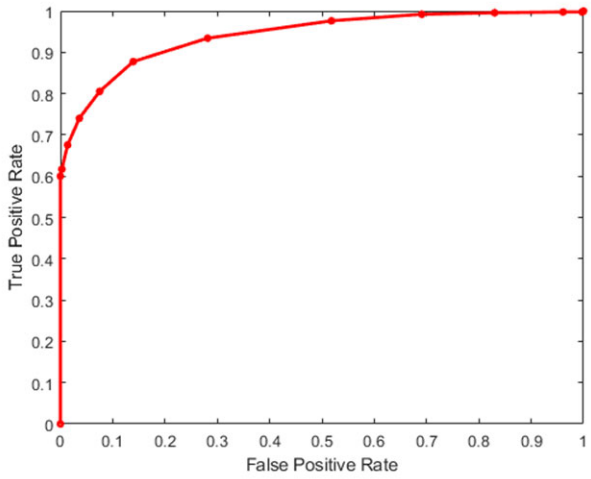
The ROC curves of the two criteria are shown in Figure 4.4. We get 96.5% AUC and 9.4% EER at the frame-level evaluation criterion, and 94.1% AUC and 12.5% EER in the pixel-level evaluation criterion. In our experiment, we select some previous traditional methods and deep learning methods for comparison. The comparison results are shown in Table 4.1. Our approach achieves the highest AUC and lower EER at both frame-level and pixel-level, which proves the capability of the multi-input and patch-based evaluation method for anomalous event detection and localization.

We prepare six sets of pictures in Figure 4.5: (a) are the original frames, (b) are the generated appearance frames, (c) are the motion frames, (d) are the generated motion

frames and (e) are the anomaly events localization results. The red area in (e) is composed of abnormal patches. These figures represent that with the help of the patch-based evaluation method, our TMGNN is able to detect and localize all possible anomalous objects including bikers, skateboarders and cars.



(a) Frame-level Criterion



(b) Pixel-level Criterion

Figure 4.4: Frame-level and Pixel-level ROC curve

Method	frame-level		pixel-level	
	EER	AUC	EER	AUC
MPPCA [31]	30%	69.3%	N/A	N/A
SR+VAE [32]	18%	89.1%	N/A	N/A
AMDN (Double Fusion) [33]	17%	90.8%	N/A	N/A
Hybrid model [34]	11.5%	88.9%	N/A	N/A
Convolutional AE [35]	21.7%	90%	N/A	N/A
3DCNN-GAN [36]	11%	93%	N/A	N/A
AbnormalGAN [37]	14%	93.5%	N/A	N/A
Future frame prediction [6]	N/A	95.4%	N/A	N/A
ConvAE-Unet [38]	N/A	96.2%	N/A	N/A
MPPCA+SF [39]	36%	61.3%	72%	N/A
Social Force (SF) [40]	42%	55.6%	80%	N/A
MDT [39]	25%	82.9%	54%	N/A
AnomalyNet [41]	10.3%	94.9%	N/A	52.8%
GMM-FCN [42]	12.6%	92.2%	19.2%	78.2%
DSTN [43]	9.4%	95.5%	21.8%	83.1%
Two-stage [44]	8.9%	96.4%	19.4%	85.9%
Spatial-temporal CNN [45]	24.4%	86%	N/A	88%
WTA-CAE [46]	8.9%	96.6%	16.9%	89.3%
Proposed method	**9.4%**	**96.5%**	**12.5%**	**94.1%**

Table 4.1: Frame level and pixel level performance of multiple traditional and deep learning anomaly events detection methods on the UCSD ped2 dataset

To better demonstrate the anomaly detection capability of our model, we build four figures (Figure 4.6, Figure 4.7, Figure 4.8 and Figure 4.9) showing the details of the detection process in the 2nd, 4th, 6th and 8th video clips respectively. Each figure is

comprised of two parts. The first one is a line chart of the number of abnormal patches vs frames, that illustrates how the number of abnormal patches varies throughout the video clip. Any frame with greater than or equal to 210 abnormal patches will be classified as abnormal. Another one is a series of anomaly event detection and localization results where we selected 4 sets of frames with different timestamps in each video clip. When any anomaly intrudes into the surveillance camera, the number of abnormal patches goes up rapidly and the anomalous object is detected and localized by our solution.

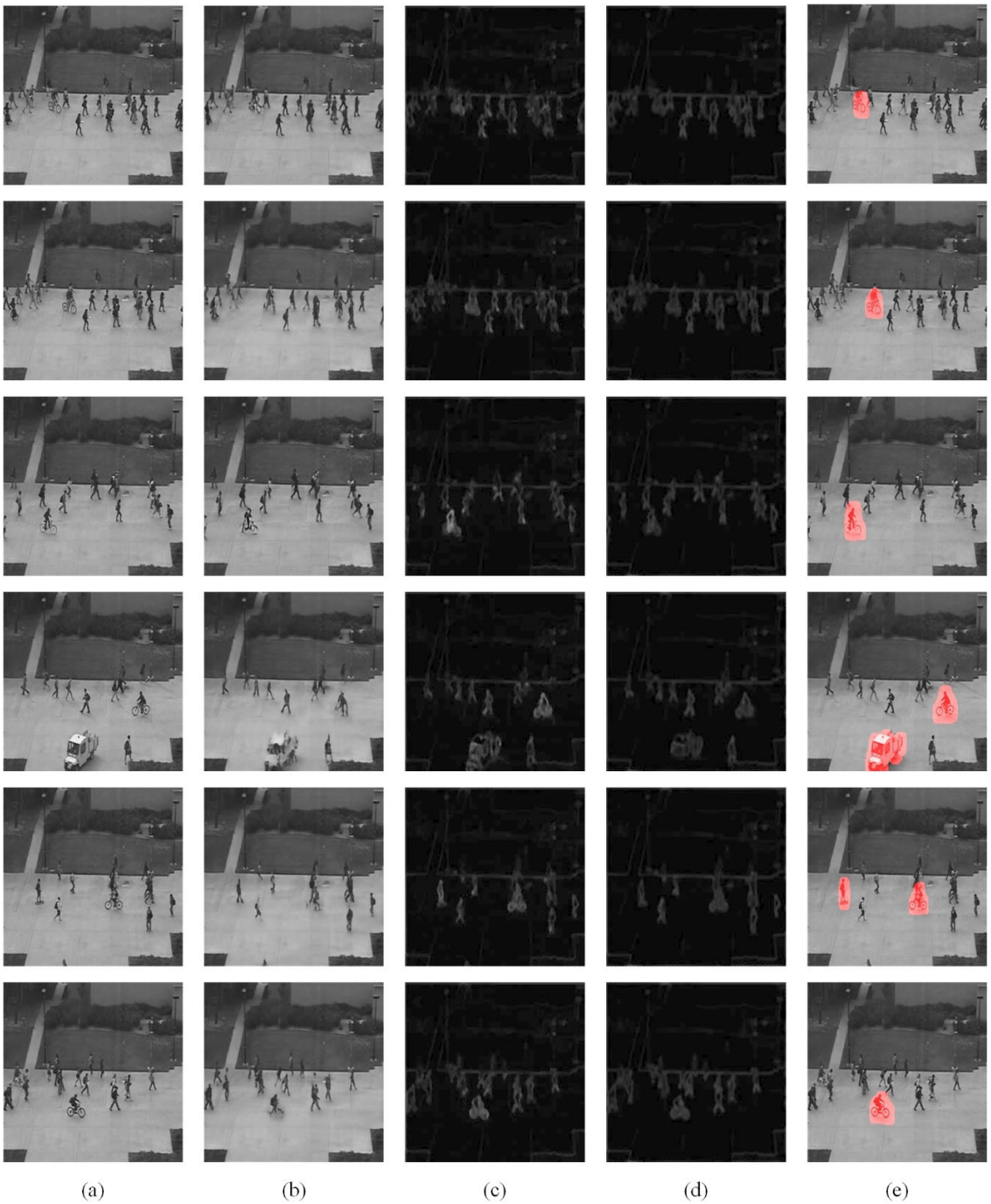


Figure 4.5: Six abnormal event detection and localization examples. (a) original frames, (b) generated appearance frames, (c) motion frames, (d) generated motion frames, (e) anomaly event localization results

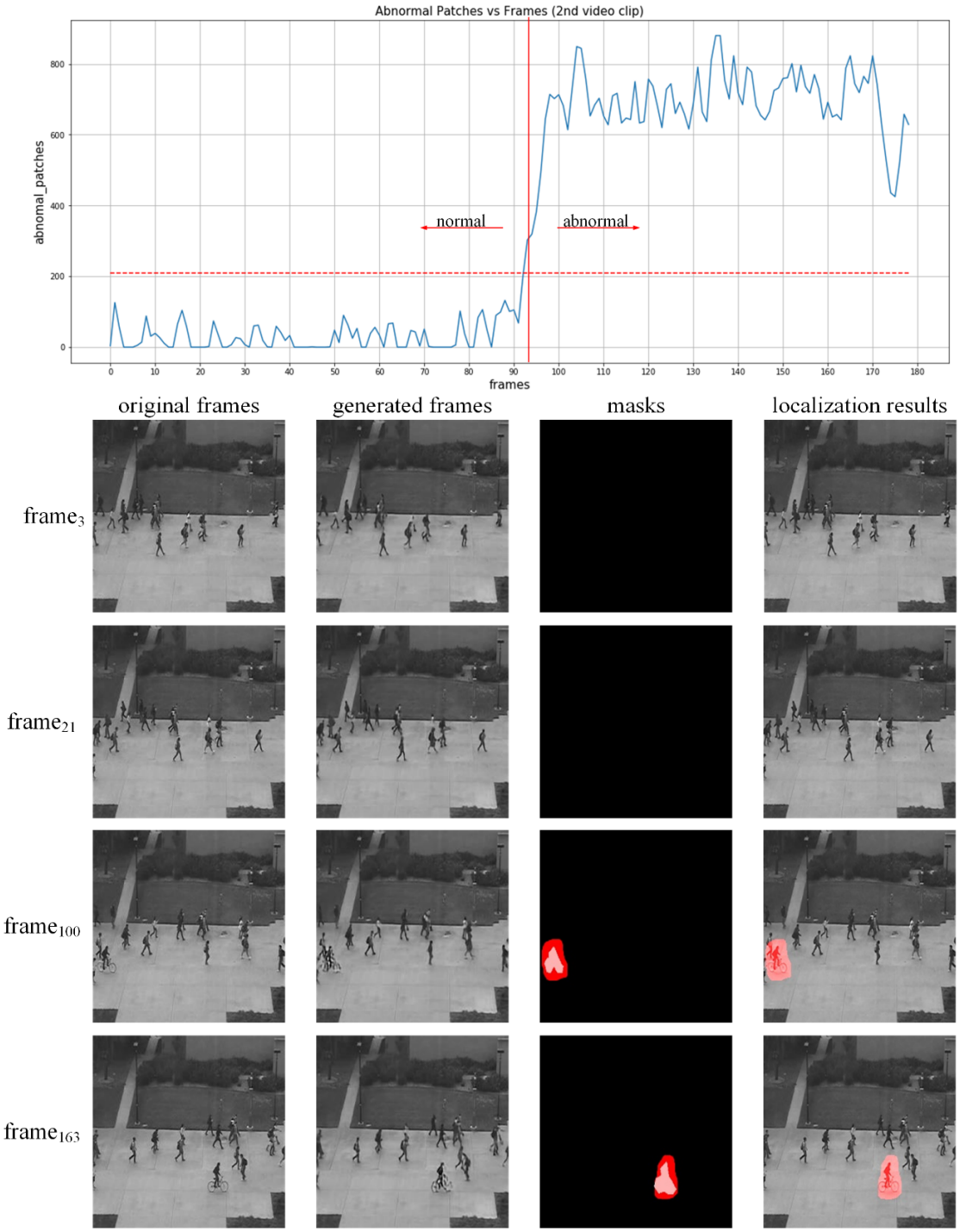


Figure 4.6: The detection process in the 2nd video clip

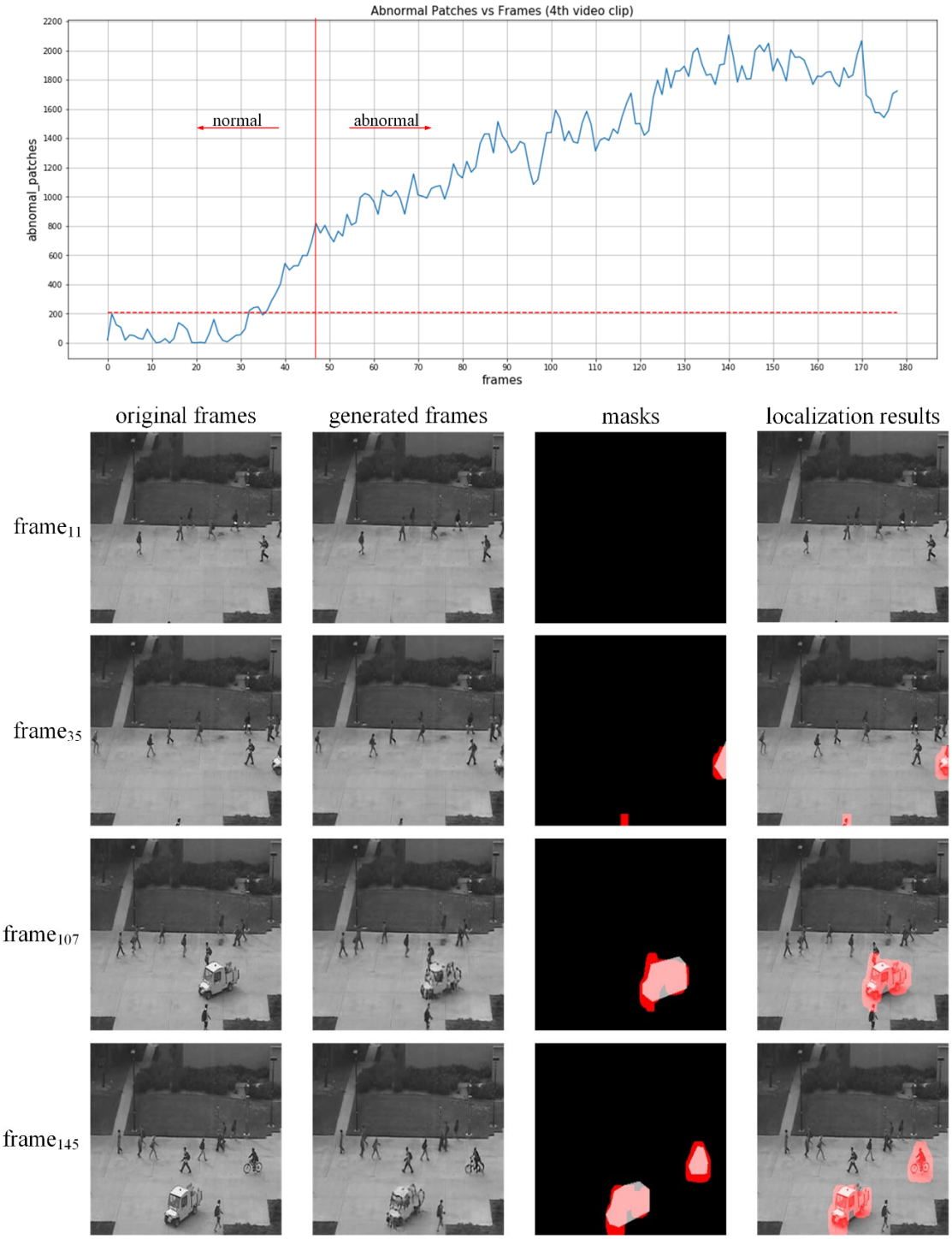


Figure 4.7: The detection process in the 4th video clip

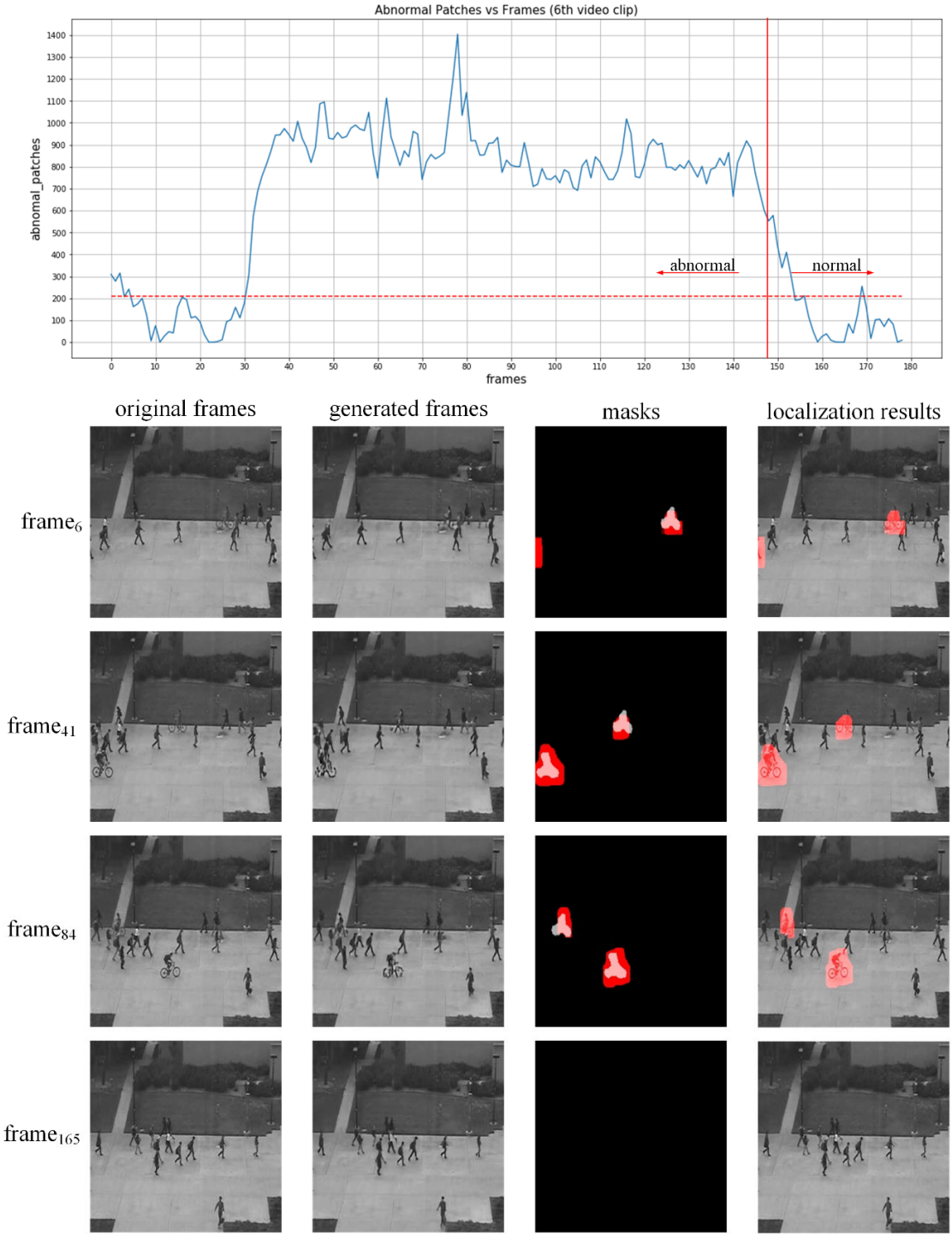


Figure 4.8: The detection process in the 6th video clip

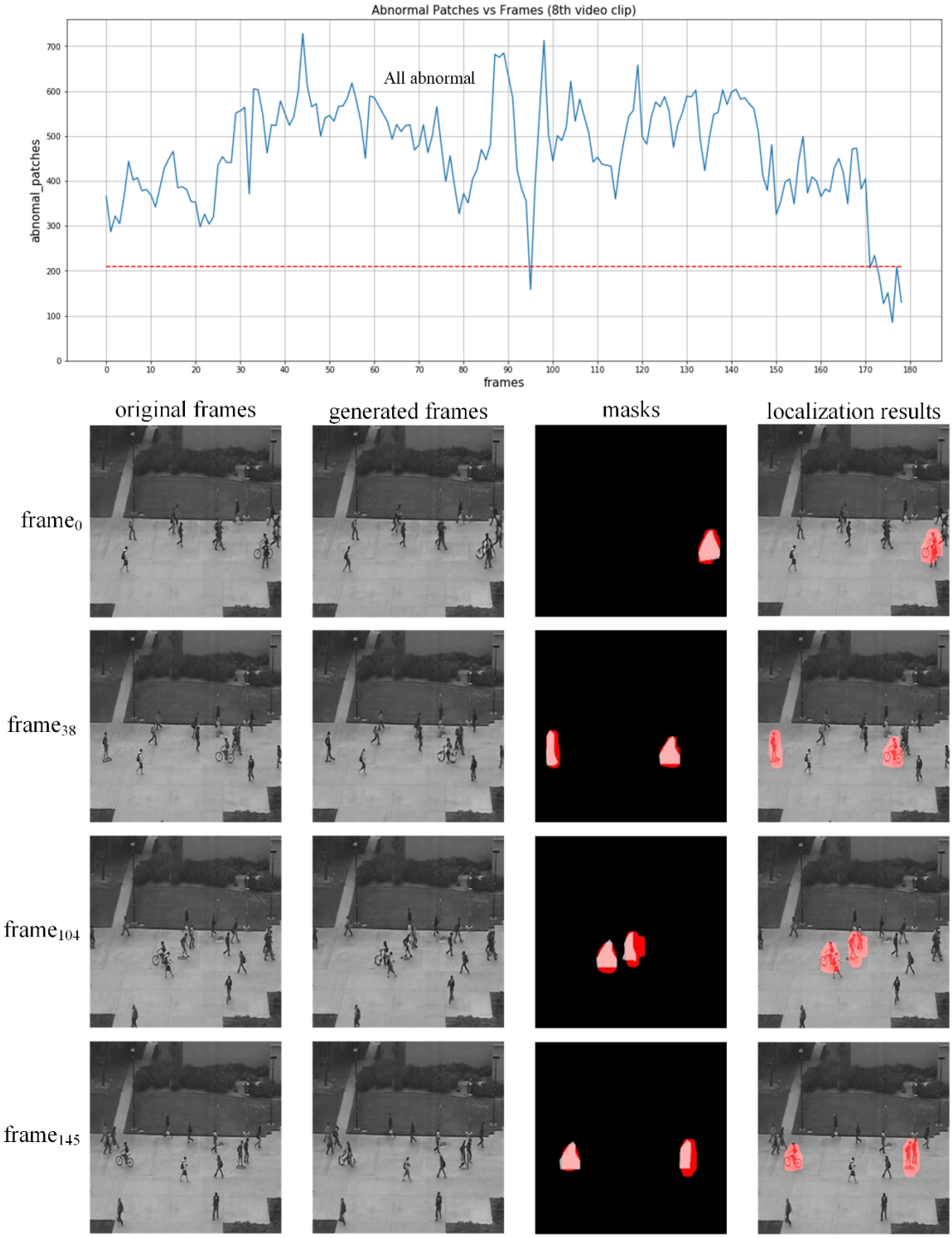


Figure 4.9: The detection process in the 8th video clip

5 Conclusion and Future Work

5.1 Conclusion

In this thesis, we propose a generative adversarial network-based framework for anomaly event detection and localization. This framework is composed of two neural networks, an appearance generation neural network and a motion generation neural network. For providing sufficient input features to the generators and improving the reconstruction ability, we design a multi-input and feed them into the networks. We decide to use the UNet with skip connections as the architecture of two generators and the discriminators are two basic binary classifiers. The two networks are trained only on normal video clips. During testing, since the two generators never learn any abnormal objects' features, we compute the absolute difference between the generated frames and the corresponding input frames, then with the help of our new patch-based evaluation method, we utilize this reconstruction error to detect and localize any abnormal object. In contrast to pixel-based and image-based evaluation methods, the patch-based method uses patches instead of pixels, which not only improves the accuracy of detecting anomalous frames but also locates any possible abnormal objects. We test our approach on the UCSD ped2 dataset.

The experiment results represent we reach 96.5% AUC and 9.4% ERR at the frame-level criterion, and 94.1% AUC and 12.5% ERR in the pixel-level criterion. Compared to state-of-the-art deep learning networks, our approach achieves higher accuracy.

5.2 Future Work

Pix2pix has already provided a good idea for generating realistic images, but it still needs paired input frames and ground truth images. This means we have to collect enough paired images for training and testing, which is quite time-consuming and infeasible. In the area of image-to-image translation, researchers propose many new neural networks based on GAN to solve this issue, such as CycleGAN [30]. Therefore, it is interesting to see if using other networks could improve the performance of video anomaly detection beyond pix2pix.

References

[1] T. Zhang, H. Lu, and S. Z. Li, "Learning Semantic Scene Models by Object Classification and Trajectory Clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogni. (CVPR)*, Miami, FL, USA, Jun. 2009, pp. 1940-1947.

[2] X. Cui, Q. Liu, M. Gao, and D. N. Metaxas, "Abnormal Detection using Interaction energy Potentials," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogni. (CVPR)*, Colorado Springs, CO, USA, Jun. 2011, pp. 3161-3167.

[3] V. Kaltsa, A. Briassouli, I. Kompatsiaris, L. J. Hadjileontiadis, and M. G. Strintzis, "Swarm Intelligence for Detecting Interesting Events in Crowded Environments," *IEEE Trans. on Image Process.*, vol. 24, no. 7, Jul. 2015, pp. 2153-2166.

[4] R. Mehran, A. Oyama, and M. Shah, "Abnormal Crowd Behavior Detection using Social Force Model," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogni. (CVPR)*, Miami, FL, Jun. 2009, pp. 935-942.

[5] Y. Cong, J. Yuan, and J. Liu, "Sparse Reconstruction Cost for Abnormal Event Detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogni. (CVPR)*, Colorado Springs, CO, USA, Jun. 2011, pp. 3449-3456.

[6] W. Liu, W. Luo, D. Lian, and S. Gao, "Future Frame Prediction for Anomaly Detection - A New Baseline," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogni. (CVPR)*, Salt Lake City, UT, Jun. 2018, pp. 6536-6545.

[7] M. Ravanbakhsh, E. Sangineto, M. Nabi, and N. Sebe, "Training Adversarial Discriminators for Cross-Channel Abnormal Event Detection in Crowds," in *IEEE Winter Conf. Appl. Comput. Vis.*, Waikoloa Village, HI, USA, Jan. 2019, pp. 1896-1904.

[8] S. Lee, H. G. Kim, and Y. M. Ro, "STAN: Spatio-Temporal Adversarial Networks for Abnormal Event Detection," in *Proc. IEEE Int. Conf. Acoust., Speech and Signal Process. (ICASSP)*, Calgary, AB, Canada, Apr. 2018, pp. 1323-1327.

[9] A. Graves, A. Mohamed, and G. Hinton, "Speech Recognition with Deep Recurrent Neural Networks," in *Proc. IEEE Int. Conf. Acoust., Speech and Signal Process. (ICASSP)*, Vancouver, BC, Canada, May 2013, pp. 6645-6649.

[10] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, Attend and Spell: A Neural Network for Large Vocabulary Conversational Speech Recognition," in *Proc. IEEE Int. Conf. Acoust., Speech and Signal Process. (ICASSP)*, Shanghai, China, Mar. 2016, pp. 4960-4964.

[11] C. Ledig et al., "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recogni. (CVPR)*, Honolulu, HI, USA, Jul. 2017, pp. 105-114.

[12] T. Tong, G. Li, X. Liu, and Q. Gao, “Image Super-Resolution Using Dense Skip Connections,” in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Venice, Italy, Oct. 2017, pp. 4809-4817.

[13] Ian J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. “Generative Adversarial Nets,” in *Proc. Neural Inf. Proc ess. Sys. (NIPS)*, Cambridge, MA, USA, Dec. 2014, pp. 2672-2680.

[14] X. J. Shi, Z. R. Chen, H. Wang, D. Y. Yeung, W. K. Wong, and W. C. Woo, “Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting,” in *Proc. Neural Inf. Process. Sys. (NIPS)*, Cambridge, MA, USA, 2015, pp. 802-810.

[15] A. B. Chan and N. Vasconcelos, “Modeling, Clustering, and Segmenting Video with Mixtures of Dynamic Textures,” *IEEE Trans. on Pattern Analysis and Machine Int elligence*, vol. 30, no. 5, May 2008, pp. 909-926.

[16] W. McCulloch and W. Pitts, “A Logical Calculus of Ideas Immanent in Nervous Act ivity,” *Bulletin of Mathematical Biophysics*, 1943, pp. 115-133.

[17] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, “Backpropagation Applied to Handwritten Zip Code Recognition,” *Neur al Computation*, vol. 1, no. 4, Dec. 1989, pp. 541-551.

[18] K. Fukushima and S. Miyake, “Neocognitron: A New Algorithm for Pattern Recognition Tolerant of Deformations and Shifts in Position,” *Pattern Recognit.*, vol. 15, no. 6, Nov.1982, pp. 455–469.

[19] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based Learning Applied to Document Recognition,” in *Proc. IEEE*, vol. 86, no. 11, Nov. 1998, pp. 2278–2324.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet Classification with Deep Convolutional Neural Networks,” in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 25, 2012, pp. 1106–1114.

[21] C. Szegedy et al., “Going Deeper with Convolutions,” in *Proc. IEEE Conf. Comput. Vis. Pattern Rcogni. (CVPR)*, Boston, MA, USA, Jun. 2015, pp. 1–9.

[22] K. Simonyan, and A. Zisserman, “Very Deep Convolutional Networks for Large-scale Image Recognition,” in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2015.

[23] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” in *Int. Conf. Learn. Represent. (ICLR)*, Las Vegas, NV, USA, Jun. 2016, pp. 770–778.

[24] M. Mirza and S. Osindero, “Conditional Generative Adversarial Nets,” 2014. [Online]. Available: https://doi.org/10.48550/arXiv.1411.1784.

[25] A. Radford, L. Metz, and S. Chintala, “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks,” in *Proc. Int. Conf. Learn. R epresent.* (ICLR), 2016.

[26] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein Generative Adversarial Netw orks,” in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2017, pp. 214-223.

[27] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-Image Translation with Con ditional Adversarial Networks,” in *Proc. IEEE Conf. Comput. Vis. Pattern Rcogni. (CVPR)*, Honolulu, HI, USA, Jul. 2017, pp. 5967-5976.

[28] K. Xu, T. Sun and X. Jiang, "Video Anomaly Detection and Localization Based on a n Adaptive Intra-Frame Classification Network," *IEEE Trans. Multimedia (MM)*, vo l. 22, no. 2, Feb. 2020, pp. 394-406.

[29] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image Quality Assessm ent: From Error Visibility to Structural Similarity,” *IEEE Trans. on Image Process.*, vol. 13, no. 4, Apr. 2004, pp. 600-612.

[30] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks,” in *Proc. IEEE Int. Conf. Comput. V is. (ICCV)*, Venice, Italy, Oct. 2017, pp. 2242-2251.

[31] J. Kim and K. Grauman, “Observe locally, infer globally: A Apace-time MRF for D etecting Abnormal Activities with Incremental Updates,” in *Proc. IEEE Conf. Comp ut. Vis. Pattern Recognit. (CVPR)*, Miami, FL, USA, Jun. 2009, pp. 2921–2928.

[32] J. Sun, X. Wang, N. Xiong, and J. Shao, “Learning Sparse Representation With Vari ational Auto-Encoder for Anomaly Detection,” *IEEE Access*, vol. 6, 2018, pp. 3335 3–33361.

[33] D. Xu, Y. Yan, E. Ricci, and N. Sebe, “Detecting Anomalous Events in Videos by L earning Deep Representations of Appearance and Motion,” *Comput. Vis. Image Und erstand.*, vol. 156, Mar. 2017, pp. 117–127.

[34] L. Wang, F. Zhou, Z. Li, W. Zuo and H. Tan, “Abnormal Event Detection in Videos Using Hybrid Spatio-Temporal Autoencoder,” in *Proc. IEEE Int. Conf. Image Proce ss. (ICIP)*, Athens, Greece, Oct. 2018, pp. 2276-2280.

[35] M. Hasan, J. Choi, J. Neumann, A. K. Roy-Chowdhury, and L. S. Davis, “Learning temporal regularity in video sequences,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 733-742.

[36] W. Shin, S. Bu and S. Cho, “3D-Convolutional Neural Network with Generative Ad versarial Network and Autoencoder for Robust Anomaly Detection in Video Surveil lance,” *Int. J. Neural Syst.*, vol. 30, no. 06, May 2020.

[37] M. Ravanbakhsh, M. Nabi, E. Sangineto, L. Marcenaro, C. Regazzoni and N. Sebe, "Abnormal Event Detection in Videos using Generative Adversarial Nets," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Beijing, China, Sep. 2017, pp. 1577-1581.

[38] T. N. Nguyen and J. Meunier, "Anomaly Detection in Video Sequence with Appearance-Motion Correspondence," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Seoul, Korea (South), Oct. 2019, pp. 1273-1283.

[39] Vijay Mahadevan, Weixin Li, Viral Bhalodia, and Nuno Vasconcelos, "Anomaly detection in crowded scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, San Francisco, CA, USA, Jun. 2010, pp. 1975–1981.

[40] Ramin Mehran, Alexis Oyama, and Mubarak Shah, "Abnormal Crowd Behavior Detection using Social Force Model," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2009, pp. 935–942.

[41] J. T. Zhou, J. Du, H. Zhu, X. Peng, Y. Liu and R. S. M. Goh, "AnomalyNet: An Anomaly Detection Network for Video Surveillance," *IEEE Trans. Inf. Forensics Secur.*, vol. 14, no. 10, Oct. 2019, pp. 2537-2550.

[42] Y. Fan, G. Wen, D. Li, S. Qiu, and M. D. Levine, "Video Anomaly Detection and Localization via Gaussian Mixture Fully Convolutional Variational Autoencoder," 2018. [Online]. Available: http://arxiv.org/abs/1805.11223.

[43] T. Ganokratanaa, S. Aramvith and N. Sebe, “Unsupervised Anomaly Detection and Localization Based on Deep Spatiotemporal Translation Network,” *IEEE Access*, vo l. 8, Mar. 2020, pp. 50312-50329.

[44] S. Wang, Y. Zeng, Q. Liu, C. Zhu, E. Zhu, and J. Yin, “Detecting Abnormality witho ut Knowing Normality: A Two-stage Approach for Unsupervised Video Abnormal E vent Detection,” in *Proc. ACM Multimed. Conf.*, Seoul, Korea (South), Oct. 2018, p p. 636-644.

[45] S. Zhou, W. Shen, D. Zeng, M. Fang, Y. Wei, and Z. Zhang, “Spatial-temporal Conv olutional Neural Networks for Anomaly Detection and Localization in Crowded Sce nes,” *Sig. Process. Image Commun.*, vol 47, 2016, pp. 358-368.

[46] H. T.M. Tran, and D. Hogg, “Anomaly Detection using a Convolutional Winner-tak e-all Autoencoder,” in *Proc. British Machine Vision Conference (BMVC)*, Sep. 2017.