sensors

MDPI

Article

Research on Anomaly Detection of Surveillance Video Based on Branch-Fusion Net and CSAM

Pengjv Zhang and Yuanyao Lu *

School of Information Science and Technology, North China University of Technology, Beijing 100144, China
* Correspondence: luyy@ncut.edu.cn

Abstract: As the monitor probes are used more and more widely these days, the task of detecting abnormal behaviors in surveillance videos has gained widespread attention. The generalization ability and parameter overhead of the model affect how accurate the detection result is. To deal with the poor generalization ability and high parameter overhead of the model in existing anomaly detection methods, we propose a three-dimensional multi-branch convolutional fusion network, named "Branch-Fusion Net". The network is designed with a multi-branch structure not only to significantly reduce parameter overhead but also to improve the generalization ability by understanding the input feature map from different perspectives. To ignore useless features during the model training, we propose a simple yet effective Channel Spatial Attention Module (CSAM), which sequentially focuses attention on key channels and spatial feature regions to suppress useless features and enhance important features. We combine the Branch-Fusion Net and the CSAM as a local feature extraction network and use the Bi-Directional Gated Recurrent Unit (Bi-GRU) to extract global feature information. The experiments are validated on a self-built Crimes-mini dataset, and the accuracy of anomaly detection in surveillance videos reaches 93.55% on the test set. The result shows that the model proposed in the paper significantly improves the accuracy of anomaly detection in surveillance videos with low parameter overhead.

Keywords: video anomaly detection; C3D network; attention mechanisms; Bi-GRU

Citation: Zhang, P.; Lu, Y. Research on Anomaly Detection of Surveillance Video Based on Branch-Fusion Net and CSAM. *Sensors* **2023**, *23*, 1385. https://doi.org/10.3390/s23031385

Academic Editor: Giovanni Betta

Received: 12 December 2022
Revised: 18 January 2023
Accepted: 23 January 2023
Published: 26 January 2023

1. Introduction

To maintain public safety, more and more surveillance probes are deployed at road intersections, shopping centers, subway entrances, and other public locations, resulting in a flood of surveillance videos [1,2]. Law enforcement officials usually spend a lot of time watching surveillance videos to find crimes, but still frequently miss important details. People urgently need an algorithm that automatically detects abnormal behaviors in surveillance video to lessen the workload of law enforcement officials while also dealing with abnormal criminal behaviors faster and more correctly [3–5].

Video anomaly detection methods can be divided into four categories: reconstruction-based method, prediction-based method, classification-based method, and regression-based method. The reconstruction-based method works by training normal video data to generate a distribution representation of normal data [6,7]. The prediction-based method assumes that a continuous normal video has a regular contextual connection and that future frames can be predicted by learning the dependencies [8]. However, there are no such dependencies between frames in abnormal videos [9]. The classification-based method treats the detection of anomalies as a classification problem [10]. After learning multiple distribution patterns of normal samples, samples that do not follow these distribution patterns are classified as abnormal. The above three categories of methods are suitable for frame-level detection of video, but not for large-scale detection [11].

Therefore, video anomaly detection is mostly performed by the regression-based method. The method's idea is to use the anomaly score as an evaluation indicator. After

setting an appropriate threshold, the video is considered abnormal if the anomaly score is above the threshold. The method is suitable for the large-scale video anomaly detection due to the simple structure. The regression-based method is employed in this paper's multiple-instance learning (as shown in Figure 1). The C3D network is used in the feature extraction network for multiple-instance learning. Each convolutional layer of the C3D network convolves the input feature map directly in three dimensions, which leads to two problems [12]. Firstly, the large size and number of 3D convolutional kernels lead to high parameter overhead [13,14]. Secondly, the C3D network understands the input features from only one perspective because of the direct convolution of the feature map, which leads to poor generalization ability.

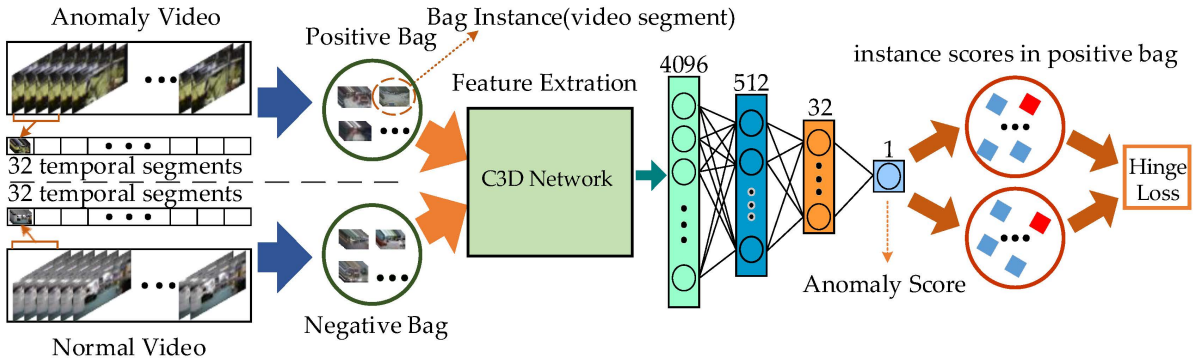

Figure 1. Multiple-instance learning. The surveillance video is divided into a fixed number of video segments, and the segments are placed in positive and negative bags. Each video segment is called an instance in the bags [15]. Then, the feature extraction network is used to extract features from the fixed number of video clips in the two bags. Next, the multilayer perceptron is used to calculate the anomaly scores of the features. Finally, the scores are used to determine whether there is abnormal behavior.

To address the problems, we propose a feature extraction network named "Branch-Fusion Net". The multi-branch structure not only reduces parameter overhead but also allows the network to understand the input feature maps from different perspectives to improve the generalization ability.

The features extracted by networks usually contain features that are not relevant to the detection task [16]. To suppress useless features and enhance important features, we propose a simple yet effective Channel Spatial Attention Module (CSAM). The CSAM is an end-to-end generic module that can be seamlessly integrated into three-dimensional convolutional neural networks.

Since the convolutional neural networks are suitable for extracting local features [17], we combine the Branch-Fusion Net and the CSAM as the local feature extraction network and use the Bi-Directional Gated Recurrent Unit (Bi-GRU) for the global feature extraction.

The contributions of our work include:

(1) We propose the Branch-Fusion Net, which not only greatly reduces parameter overhead, but also improves the generalization ability of feature extraction by understanding the input feature maps from multiple perspectives. The network achieves state-of-the-art performance for behavior recognition tasks on multiple benchmarks.
(2) We propose a simple yet effective CSAM to ignore useless features during the model training. The CSAM focuses attention on key channels and spatial feature areas in turn. Adding the CSAM to the mainstream 3D convolutional neural networks can significantly improve the feature extraction effect.
(3) We establish a surveillance video dataset called 'Crimes-mini' containing five categories of abnormal behaviors. The model we propose achieves the detection accuracy of 93.55% on the test set.

2. Related Work

2.1. Anomaly Detection Methods

There are four methods to deal with anomaly detection, including the reconstruction-based method, prediction-based method, classification-based method, and regression-based method. Table 1 shows the advantages and disadvantages of the four methods.

Table 1. Comparison and summary of different kind of methods.

Category	Judgment Basis	Advantages	Disadvantages
Reconstruction-based method	Small reconstruction errors for normal video frames and large reconstruction errors for abnormal video frames.	High accuracy in detecting location-related anomalies.	There are reconstruction errors for anomalous video frames.
Prediction-based method	Small prediction errors for normal video frames and large prediction errors for abnormal video frames.	High accuracy in detecting motion-related anomalies.	It ignores the fact that normal video frames can be unpredictable.
Classification-based method	The samples that do not follow the normal sample distribution are considered abnormal.	The distribution of normal samples can be well-learned.	If the normal sample distribution is complex, the classification model may fail.
Regression-based method	The samples with anomaly scores above the threshold are considered abnormal.	The model is simple and suitable for large-scale video anomaly detection.	The threshold for video anomalies is not easy to determine.

Reconstruction-based method. Hasan et al. propose two Auto-Encoder-based methods that can perform anomaly detection without supervision [18], but the abnormal samples are sometimes subject to reconstruction errors. To address the shortcoming, Gong et al. propose an improved Auto-Encoder [19], named the "Memory-augmented Auto-Encoder (MemAE)". The method obtains the encoding from the encoder based on the input, and then uses it as a query to retrieve the most relevant memory item for reconstruction. To better remember normal samples, Park et al. propose a memory module that can be updated according to the scheme [20].

Prediction-based method. Medel et al. propose the Convolutional Long Short-Term Memory Network (Conv-LSTM) [21], which incorporates convolution operation into the Long Short-Term Memory Network (LSTM). Anomaly detection is achieved by reconstructing past frames and predicting future frames. Lu et al. combine the Variational Auto-Encoder (VAE) with the Conv-LSTM to propose the Convolutional Variational Recurrent Neural Network (Conv-VRNN) for generating future frames [22]. To address the possible blurring of future frames, Mathieu et al. [23] use alternating convolution and Rectified Linear Unit (RLU) to generate future frames and propose a method to fuse multiple feature learning strategies to generate clear future frames. Ye et al. propose a novel deep Predictive Coding Network (AnoPCN) [24], which implements anomaly detection by unifying reconstruction and prediction methods into a framework.

Classification-based method. Sabokrou et al. [25] propose a single classification-based video anomaly detection method inspired by the Generative Adversarial Network (GAN) for training models. Based on Sabokrou's study, Wu et al. propose a deep single classification neural network that is capable of obtaining compact single-class classifiers considering only normal samples [26]. Xu et al. propose an adaptive intra-frame classification network [27]. The network extracts and classifies the input appearance and motion features into several sub-regions, and then classifies the sub-regions. If the test classification result of the sub-region is different from the true classification, the video is considered abnormal.

Regression-based method. Sultani et al. propose an anomaly detection method based on multiple-instance learning by scoring abnormal and normal video clips and then picking out the clips with the highest anomaly scores to train the model [28]. Since the hinge loss function used for multiple-instance learning training is not smooth, Kamoona et al. propose a loss function to make the model robust to output anomaly scores [29]. Zhu et al.

output anomaly scores by feeding the calculated optical flow into a sequence-enhancement network [30].

Among the four methods, only the regression-based method is suitable for dealing with large-scale video data, which can be applied in practical scenarios. Therefore, we use multiple-instance learning of the regression-based method for anomaly detection. With the multiple-instance learning for anomaly detection, how to efficiently extract the features is crucial to the detection result.

2.2. Feature Extraction Network for Video

The networks for feature extraction in videos can be divided into a two-stream network architecture and a three-dimensional convolutional neural network architecture.

The three-dimensional convolutional neural network architecture was firstly proposed in [12]. The 3D convolution adds a temporal dimension to the 2D convolution to extract features in both temporal and spatial dimensions. The output of the 3D convolution is still a 3D feature map. Specifically, the 3D convolution moves in three dimensions: height, width, and depth, for multiple video frames. At each position, element-by-element multiplication and addition provide a value. Since the filter is sliding through 3D space, the output values are also arranged in 3D space. The C3D network is shown in Figure 2, and the size of each layer is shown in Table 2.

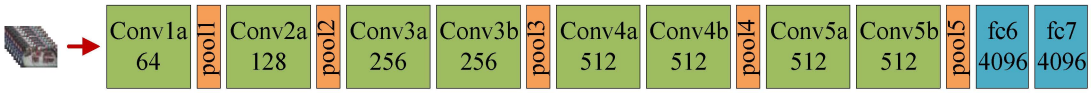

Figure 2. C3D network architecture.

Table 2. The size of each layer in C3D network.

Layer Name	Size	Stride
Conv1a	$3 \times 3 \times 3$	$1 \times 1 \times 1$
pool1	$1 \times 2 \times 2$	$1 \times 2 \times 2$
Conv2a	$3 \times 3 \times 3$	$1 \times 1 \times 1$
Pool2	$2 \times 2 \times 2$	$2 \times 2 \times 2$
Conv3a	$3 \times 3 \times 3$	$1 \times 1 \times 1$
Conv3b	$3 \times 3 \times 3$	$1 \times 1 \times 1$
pool3	$2 \times 2 \times 2$	$2 \times 2 \times 2$
Conv4a	$3 \times 3 \times 3$	$1 \times 1 \times 1$
Conv4b	$3 \times 3 \times 3$	$1 \times 1 \times 1$
pool4	$2 \times 2 \times 2$	$2 \times 2 \times 2$
Conv5a	$3 \times 3 \times 3$	$1 \times 1 \times 1$
Conv5b	$3 \times 3 \times 3$	$1 \times 1 \times 1$
pool5	$2 \times 2 \times 2$	$2 \times 2 \times 2$

The three-dimensional convolutional neural network architecture can directly utilize three-dimensional convolution to extract spatiotemporal features, so the network structure is simple. However, the increase in the dimensionality of 3D convolution leads to a large number of parameters in the network.

The two-stream architecture (as shown in Figure 3) was firstly proposed in [31]. Spatial Stream Convnet and Temporal Stream Convnet are used to extract the spatial and temporal features of the video, respectively, and finally, the two networks are fused by late fusion. Specifically, the input of Spatial Stream Convnet is a single video frame, which is responsible for extracting the appearance features of the frames. Multiple optical stream frames are input into Temporal Stream Convnet. The addition of an optical stream makes it easier to capture motion information in the neural network, and directly provides the model with motion information between frames. However, the optical flow needs to be pre-extracted, and it leads to the spatiotemporal feature extraction being time-consuming.

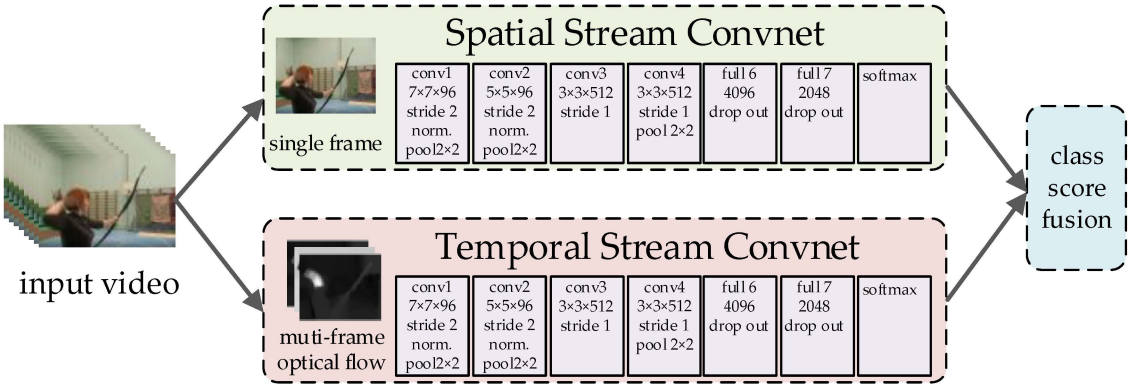

Figure 3. Two-stream network architecture.

3. The Network Architecture

To introduce our model, we show the architecture in Figure 4, and a summary of the size of each stage is shown in Table 3.

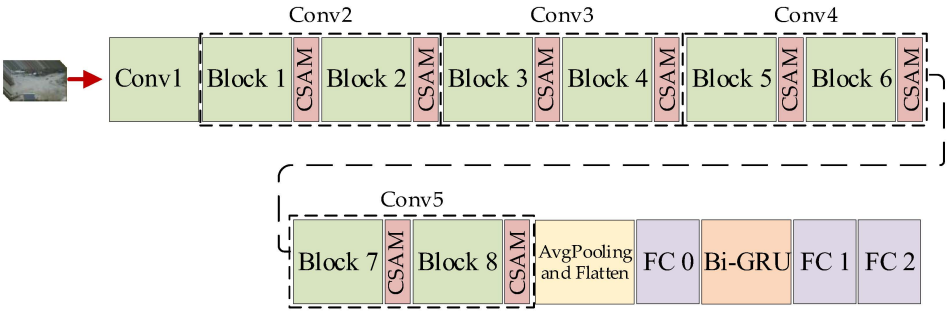

Figure 4. The architecture of our model.

Table 3. The size of each stage in our model.

Stage	Output Size	The Size of the Stage
Conv1	$64 \times 8 \times 28 \times 28$	$3 \times 3 \times 3, 64$
Conv2	$256 \times 8 \times 28 \times 28$	$\begin{bmatrix} 1 \times 1 \times 1, 128 \\ 3 \times 3 \times 3, 128, C = 32 \\ 1 \times 1 \times 1, 256 \end{bmatrix} \times 2$
Conv3	$512 \times 4 \times 14 \times 14$	$\begin{bmatrix} 1 \times 1 \times 1, 128 \\ 3 \times 3 \times 3, 128, C = 32 \\ 1 \times 1 \times 1, 512 \end{bmatrix} \times 2$
Conv4	$1024 \times 2 \times 7 \times 7$	$\begin{bmatrix} 1 \times 1 \times 1, 256 \\ 3 \times 3 \times 3, 256, C = 32 \\ 1 \times 1 \times 1, 1024 \end{bmatrix} \times 2$
Conv5	$2048 \times 1 \times 4 \times 4$	$\begin{bmatrix} 1 \times 1 \times 1, 512 \\ 3 \times 3 \times 3, 512, C = 32 \\ 1 \times 1 \times 1, 2048 \end{bmatrix} \times 2$
AvgPooling and Flattening	1×2048	
FC0	1×300	2048×300
Bi-GRU	$1 \times 1 \times 256$	Hidden_size = 128 Num_layer = 2
FC1	1×128	256×128
FC2	1×4096	128×4096

The Branch-Fusion Net consists of five stages: Conv1, Conv2, Conv3, Conv4, and Conv5. We combined the Branch-Fusion Net and the CSAM as the local feature extrac-

tion network and used the Bi-Directional Gated Recurrent Unit (Bi-GRU) for the global feature extraction.

3.1. Branch-Fusion Net

We noted that group convolution not only reduces the parameter overhead but also understands the input features from different perspectives after dividing different channels into groups, so we proposed the multi-branch structure based on group convolution. We firstly introduce how group convolution reduces the parameter overhead.

For the feature map of C × D × H × W, we can generate N × D × H × W by using N convolution kernels of C × d × h × w (assuming that the image size remains the same after pooling). Then, we assumed that the size of the input feature map was still C × D × H × W (Channel × Depth × Height × Width) and the number of output feature maps was N. If we divide it into 2 groups (as shown in Figure 5), the size of the input feature map is C/2 × D × H × W for each group, and the number of output feature maps of each group is N/2. The size of each convolution kernel is C/2 × d × h × w. The number of convolutional kernels in each group is N/2. The kernels only convolve with the input feature map of the same group. Therefore, the total number of convolutional kernels was N × C/2 × d × h × w. It can be seen that the total number of parameters was reduced to 1/2 of the original. When the number of groups is G, the number of parameters is reduced to 1/G.

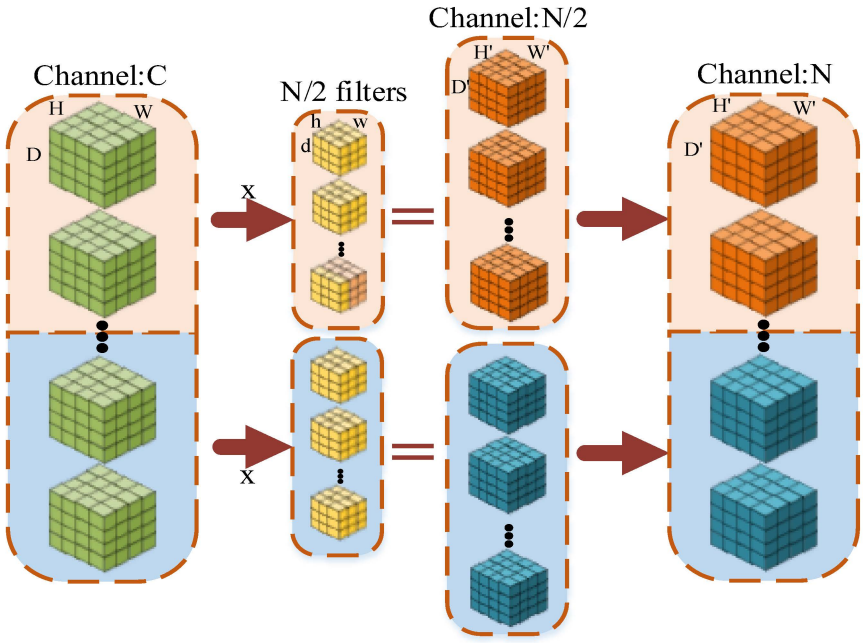

Figure 5. Group convolution diagram when the number of groups was 2.

Since each group in group convolution understands the input features independently and information does not circulate among different groups, this leads to the fact that each branch will understand the features from a different perspective.

Since group convolution groups input features from channel dimensions, which is not conducive to extracting global channel features, we improved group convolution by proposing the multi-branch structure, as shown in Figure 6. The first convolutional layer consisted of 128(4 × 32) convolutional kernels of 1 × 1 × 1. The number of convolution kernels was greatly reduced compared to the number of channels of input features. The size of the input feature map of the Branch-Fusion Net block was 256 × 16 × 64 × 64 (Channel × Depth × Height × Width), and the output feature map of each group was 4 × 16 × 64 × 64. We used 1 × 1 × 1 convolution for global feature preservation and completed the dimensionality reduction by making the total number of all grouped convolution kernels

smaller than the number of channels in the input feature map, which greatly reduced the number of parameters of the network.

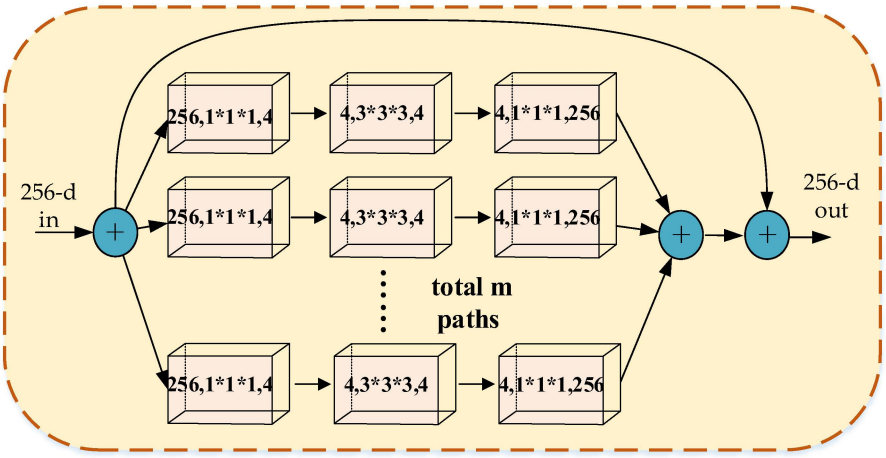

Figure 6. Block 2 of Branch-Fusion Net.

The second layer is the group convolution layer, which consisted of 128(4×32) convolution kernels of $3 \times 3 \times 3$. The number of channels in the input feature map is the same as the number of convolution kernels. In this layer, we performed group convolution of the input feature maps with branches as groups. The feature map of $128 \times 16 \times 64 \times 64$ was divided into 32 groups from the channel dimension, and each group had a feature map of $4 \times 16 \times 64 \times 64$. Each group of input features was convolved by $3 \times 3 \times 3$ to obtain an output feature map of $4 \times 16 \times 64 \times 64$. By group convolution, we further reduced the number of parameters of the network and improved the generalization ability of the network by understanding the input feature from multiple perspectives.

3.2. Channel-Spatial Attention Module

The CSAM includes the Channel Attention Module (CAM) and the Spatial Attention Module (SAM), as shown in Figure 7. The CAM is used to focus attention on the channels that have a greater impact on the final result at different times, and the SAM is used to focus attention on the regions of temporal and spatial features that are favorable for classification, which is complementary to the CAM. For the intermediate feature map of three-dimensional convolutional neural networks, the CSAM generates two attention maps along two independent dimensions, the channel and spatial, and then multiplies the attention by the input feature maps to perform adaptive feature refinement on the input feature maps.

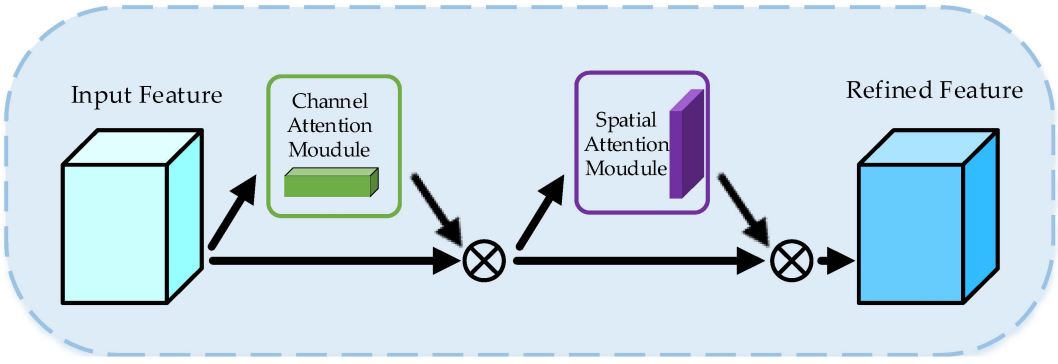

Figure 7. Channel Spatial Attention Module.

3.2.1. Channel Attention Module

In the convolutional neural network, each convolutional kernel has a different impact on the features. We used channel attention to focus our attention on the channel that has a great impact on the final result at different times. The CAM consists of two pooling layers, a multilayer perceptron, and an activation function. The structure is shown in Figure 8.

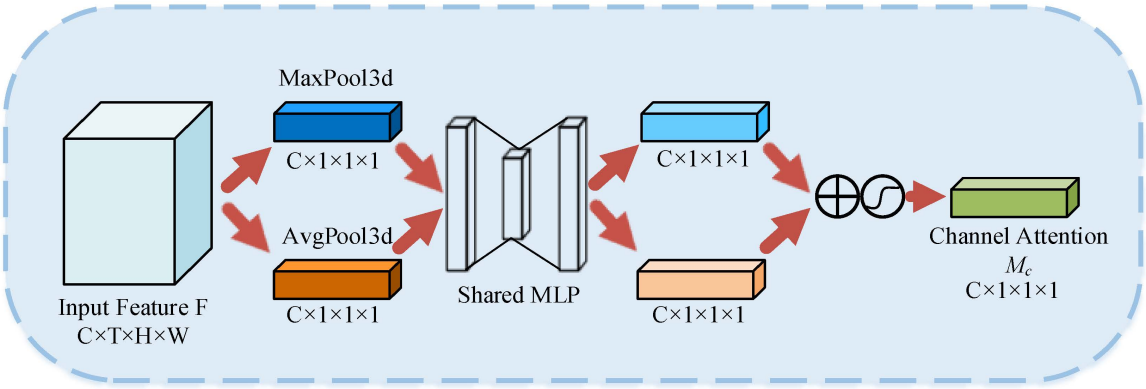

Figure 8. Channel Attention Module.

In Figure 8, $\oplus$ denotes the bitwise summation and σ denotes the sigmoid activation function. Firstly, the spatiotemporal information of the input features was aggregated using three-dimensional average pooling and three-dimensional maximum pooling to generate two different spatiotemporal context descriptors. Then, the descriptors were fed into a multilayer perceptron with shared weights to obtain two feature maps. Finally, the two feature maps were summed element-by-element and activated by the sigmoid function to obtain the final channel attention weights. The CAM is computed as:

$$M_c(F) = \sigma(MLP(AvgPool3d(F)) + MLP(MaxPool3d(F))) \quad (1)$$

where *MLP* denotes the two-layer neural network and the sigmoid activation function. Since the input was a video sequence containing temporal information, the CAM was three-dimensional. The input was changed from (batchsize, channel, height, width) to (batchsize, channel, time_sequential, height, width).

3.2.2. Spatial Attention Module

The SAM was designed to focus attention on regions of spatiotemporal features that are favorable for classification, and it complements the CAM. The SAM consists of two pooling layers, a convolutional layer, and an activation function. The structure is shown in Figure 9.

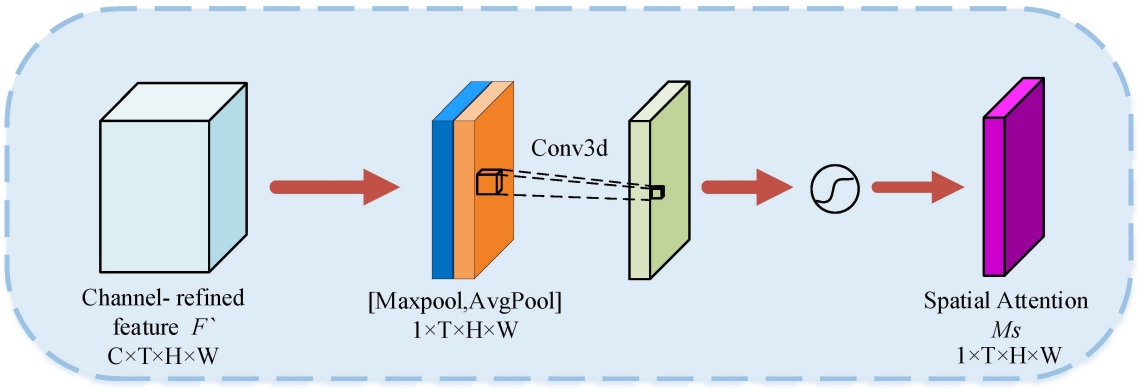

Figure 9. Spatial Attention Module.

Firstly, the channel information of the input features was aggregated using average pooling and maximum pooling to generate two different channel context descriptors. Then, the two different channel context descriptors were stitched together, and the information was aggregated by the convolutional layer with the convolutional kernel of $7 \times 7 \times 7$. Finally, the attention weights were obtained by activating the sigmoid function. The spatial attention is calculated as:

$$\mathrm{M}_s(F) = \sigma(f^{7\times7\times7}([AvgPool(F); MaxPool(F)])) \quad (2)$$

where $f^{7\times7\times7}$ denotes the convolution kernel of $7 \times 7 \times 7$ and σ denotes the sigmoid activation function. Similar to the CAM, the average pooling, maximum pooling, and convolution in spatial attention were three-dimensional.

3.3. Bi-GRU

We first introduce the Gated Recurrent Unit (GRU), as shown in Figure 10b.

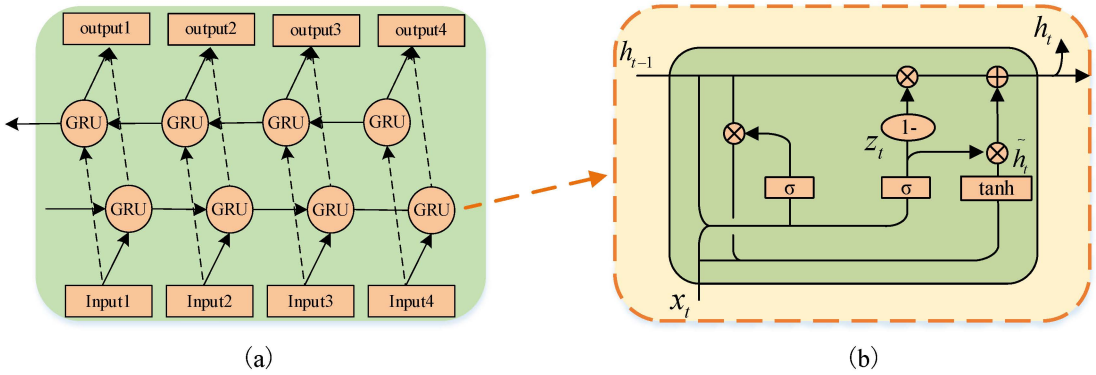

Figure 10. (**a**) The structure of Bi-GRU; (**b**) The structure of GRU.

The calculation process in the GRU is summarized as follows:

$$z_t = \sigma(W_z \bullet [h_{t-1}, x_t]) \quad (3)$$

$$r_t = \sigma(W_r \bullet [h_{t-1}, x_t]) \quad (4)$$

$$\widetilde{h_t} = \tanh(W \bullet [r_t * h_{t-1}, x_t]) \quad (5)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \widetilde{h_t} \quad (6)$$

The Gated Recurrent Unit (GRU) uses two gate functions to ensure that important features are not lost after long-term propagation. z_t is the update gate responsible for controlling the amount of data that can be saved to the current moment in the forward parameter information, and r_t is a reset gate to control how much of the past information to forget.

The idea of the Bi-Directional Gated Recurrent Unit (Bi-GRU) is to ensure the features obtained at time t have both the past and the future information. The network was divided into two independent GRUs, and the input sequences were input to the two GRUs in forward and reverse order for feature extraction. The two output vectors were stitched together to form the final feature representation. The structure of the Bi-GRU is shown in Figure 10a.

4. Experiment

4.1. Dataset

UCF-Crimes is a large dataset containing 128 h of surveillance video. It includes 1900 surveillance videos, divided into 14 categories of behavior. There exist some serious problems in the dataset, such as the low resolution and repetition of some video frames.

In this paper, we selected five categories in UCF-Crimes: arson, burglary, explosion, road accident, and stealing, and established the Crimes-mini dataset after the preprocessing operations, including cropping, de-duplication, regrouping, and expansion. The training set, validation set, and test set were randomly selected in the ratio of 6:2:2. Some of the sample images in the dataset are shown in Figure 11. The number of videos of each anomaly in our dataset is shown in Table 4.

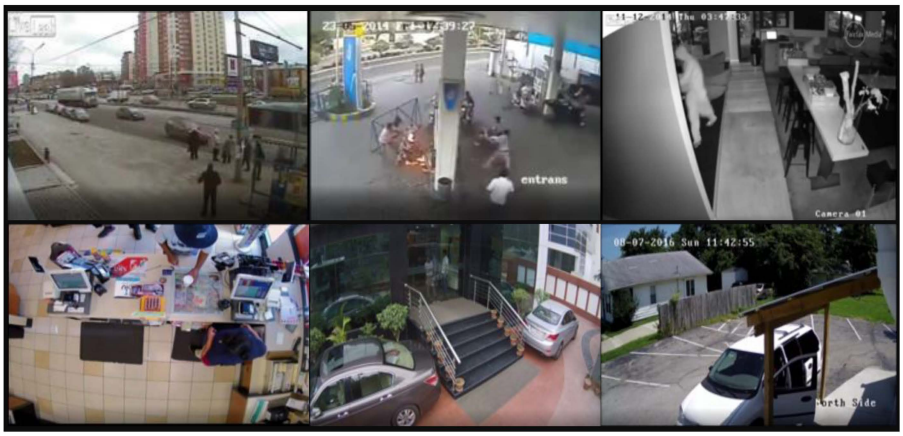

Figure 11. Sampling images with Crimes-mini dataset.

Table 4. Number of videos of each anomaly in our dataset.

Anomaly	No. of Videos
Arson	100
Burglary	100
Explosion	100
Road accident	150
Stealing	120
Normal events	320

4.2. Evaluation Index

In this paper, we chose the accuracy rate as the important evaluation metric for our experiments. Since the overall performance of the model is also critical, we considered parameter overhead and F1 values in our analysis. The accuracy and F1 values are shown in Figure 12.

Reference/Prediction	Positive	Negative
True	TP	TN
False	FP	FN

Figure 12. Confusion Matrix.

Accuracy is calculated as:

$$acc = \frac{TP + TN}{TP + TN + FN + TN} \times 100\% \quad (7)$$

$F1$ value is calculated as:

$$F1 = 2 \bullet \frac{Precise \bullet Recall}{Precise + Recall} \tag{8}$$

Recall is calculated as:

$$Recall = \frac{TP}{TP + FN} \tag{9}$$

Precise is calculated as:

$$Precise = \frac{TP}{TP + FP} \tag{10}$$

Parameter overhead is used to measure the size of the model. The lower the parameter overhead, the fewer parameters required to save the model. Parameter overhead of the 3D convolutional layer is calculated as:

$$params = C_0 \times (k_w \times k_h \times k_d \times C_i + 1) \tag{11}$$

where C_0 denotes the number of output channels, C_i denotes the number of input channels, k_w denotes the convolutional kernel width, k_h denotes the convolutional kernel height, k_d denotes the length of the convolutional kernel in the time dimension, +1 denotes bias, brackets denote the number of parameters of a convolutional kernel, and C_0 denotes that the layer has C_0 convolutional kernels.

Parameter overhead of the fully connected layer is calculated as:

$$params = (I + 1) \times O = I \times O + O \tag{12}$$

where $I \times O$ denotes the number of weights of O layers, and the number of bias is O.

4.3. Hyperparameter Setting

The setting of hyperparameters is critical, and it directly affects how well the model is trained. Therefore, we experimented with the setting of hyperparameters on the validation set.

4.3.1. Number of Branches

We used the Branch-Fusion Net as the local feature extraction network. The generalization ability of the network directly affects our detection accuracy, and the number of branches determines the generalization ability. Therefore, we set the number of branches to 16, 24, 32, and 48, and analyzed the effects of different branch numbers on the generalization ability. The experimental result for different branch numbers is shown in Figure 13.

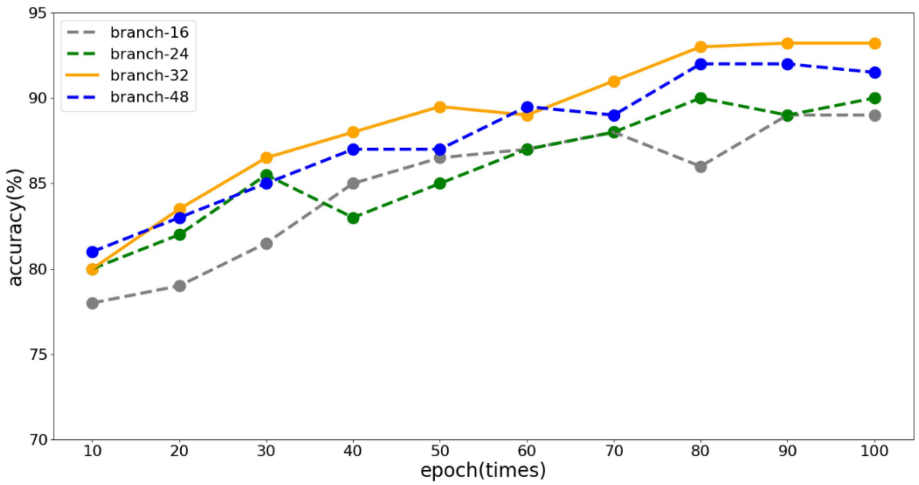

Figure 13. The effect of different numbers of branches on the accuracy rate.

The experimental result shows that the detection accuracy was the highest when the number of branches was 32. When the number of branches was 16 or 24, the lack of generalization ability of the network led to low accuracy due to the small number of branches. When the number was 48, the accuracy was slightly lower than 32 branches, and the parameter overhead was the largest. Therefore, we chose 32 as the number of branches.

4.3.2. Learning Rate Setting

The learning rate is an important hyperparameter when we train the model. For the gradient descent method, the choice of learning rate is critical. If it is too large, it will cause the model to fail to converge, and if it is too small, it will converge too slowly. We conducted experiments at different learning rates, and the experimental result is shown in Figure 14.

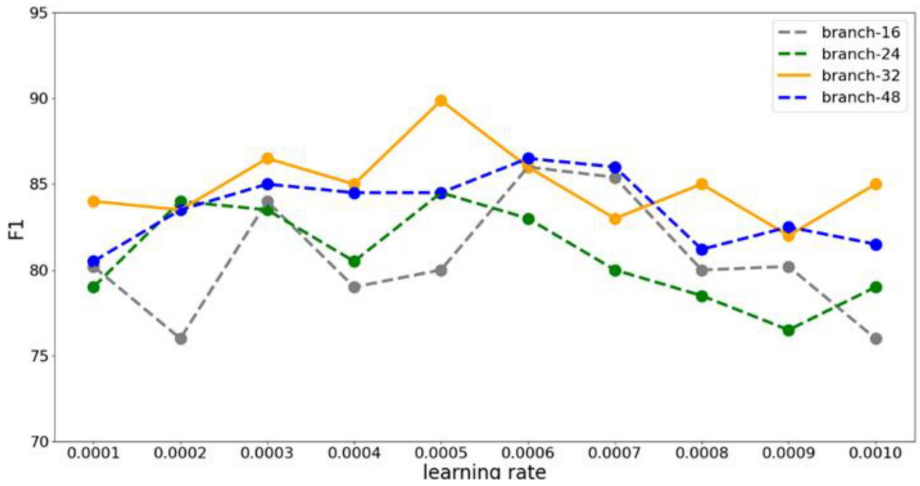

Figure 14. The effect of different learning rates on F1 values.

As can be seen from the figure, when the number of branches was 32, the F1 value was generally higher than in the other three cases, which is consistent with the results of our previous experiment. The accuracy rate and F1 value reached the maximum when the number of branches was 32 and the learning rate was 0.0005, which were 93.22% and 89.87%, respectively.

4.3.3. Number of Layers of Bi-GRU

We used the Bi-GRU to accomplish the global feature extraction, so its performance is also critical to the overall model. We trained Bi-GRU on the test set for 60 rounds in different layers. The result is shown in Figure 15.

The experimental results show that global features extracted using the single-layer network are not enough, while the features extracted by the three layers are too abstract. Both will interfere with the subsequent training and make it less effective. When Bi-GRU used two layers, it could effectively complete the extraction of global complex information.

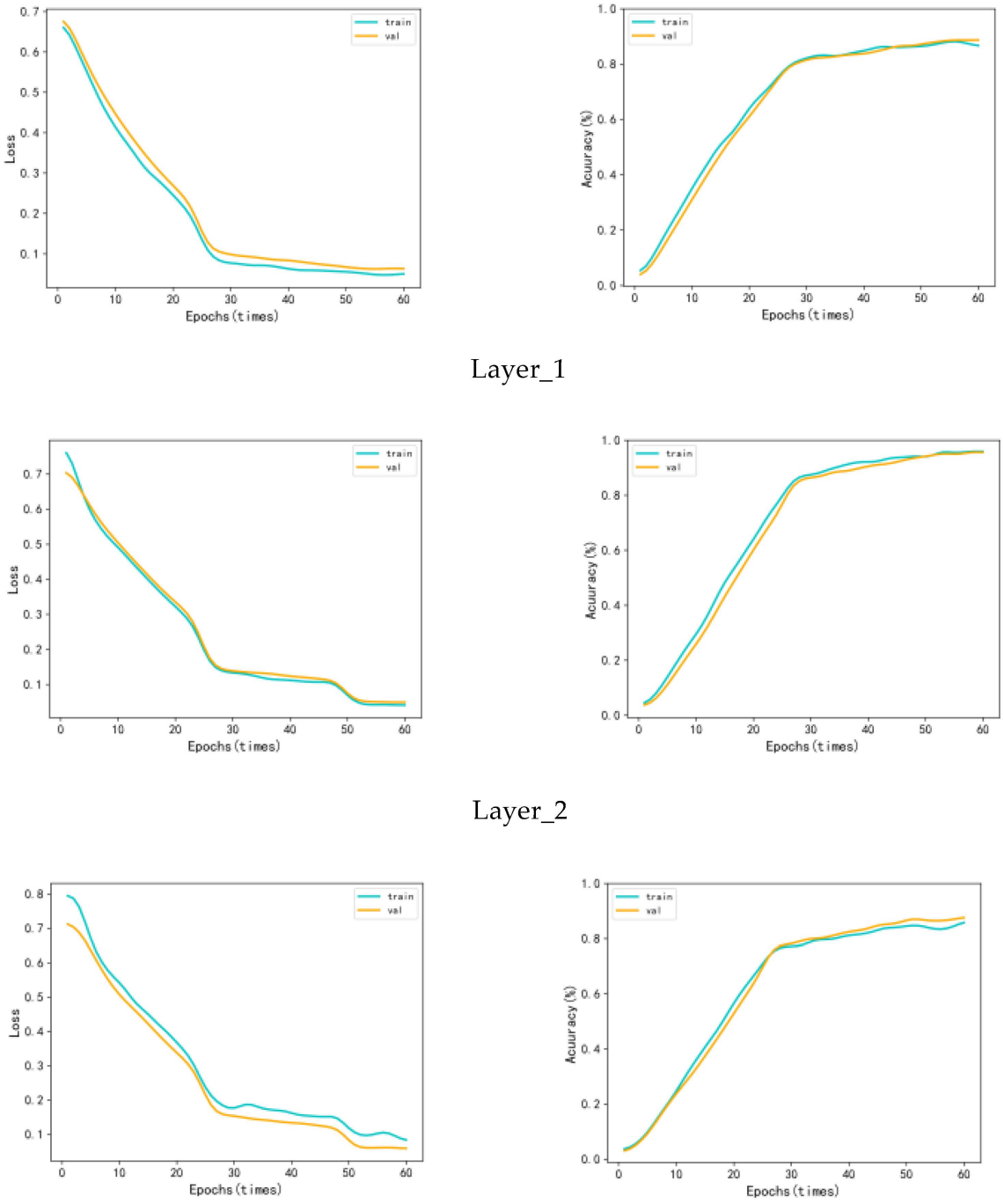

Figure 15. The training effect of Bi-GRU with different numbers of layers.

4.4. Structure Setting of CSAM

The CSAM includes two modules, the CAM and the SAM. To find the optimal connection for these two modules, we experimented with behavior recognition tasks with different connections using Branch-Fusion Net on the test set, and the experimental result is shown in Table 5.

According to the data in the table, adding either the CAM or the SAM to Branch-Fusion Net alone will improve the performance of the network, but the combination of the two is more effective than using individual modules. After connecting the two in series and parallel, we found that the "CAM + SAM" series connection was the most effective.

Table 5. Combining methods of channel and spatial attention.

Description	Accuracy (%)
Model (Branch-Fusion Net)	86.29
model + CAM	89.56
model + SAM	90.08
model + CAM and SAM in parallel	93.57
model + SAM + CAM	92.83
model + CAM + SAM	93.43

4.5. Comparison Experiments with Branch-Fusion Net

To evaluate the generalization capability of Branch-Fusion Net, we reproduced the mainstream feature extraction networks on the multiple benchmarks (UCF-101, HMDB51, and Kinetics) to conduct a comparison experiment for the behavior recognition task [32]. The Kinetics dataset is much larger than the UCF-101 dataset and the HMDB51 dataset, so it is more challenging. Therefore, we removed the networks (two-stream, C3D, and R3D) with poor feature extraction ability on the UCF-101 dataset and HMDB51 dataset, and then showed the TOP-1 accuracy and TOP-5 accuracy of the remaining networks on the Kinetics dataset. The experimental result is shown in Tables 6 and 7.

Table 6. Accuracy (%) comparison on UCF-101 and HMDB datasets.

Architecture	UCF-101	HMDB51
Two-Stream [31]	87.84	58.86
TSN [33]	93.90	70.88
TSM [34]	95.47	73.95
TEA [35]	96.63	73.12
TDN [36]	95.39	76.26
SlowFast [37]	96.20	78.04
C3D [12]	85.08	56.00
R3D [38]	85.22	53.80
R(2 + 1)D [39]	95.55	73.85
S3D [40]	96.62	75.33
X3D [41]	96.71	81.67
Two-Stream I3D [32]	97.29	80.71
NL I3D [42]	96.87	80.16
Branch-Fusion Net	97.32	82.14

Table 7. Accuracy (%) comparison on Kinetics dataset.

Description	TOP-1	TOP-5
TSN [33]	68.93	87.84
TSM [34]	74.39	90.89
TEA [35]	76.20	92.27
TDN [36]	76.85	93.16
S3D [40]	74.69	93.24
X3D [41]	79.06	93.76
R(2 + 1)D [39]	74.28	91.49
SlowFast [37]	77.11	92.37
Two-Stream I3D [32]	72.02	89.89
NL I3D [42]	76.38	92.55
Branch-Fusion Net	80.03	93.81

The experimental result shows that our proposed Branch-Fusion Net can extract features better than the mainstream networks with two-Stream architecture (two-stream, TSN, TSM, TEA, TDN, and SlowFast) and three-dimensional convolutional architecture (C3D, R3D, R(2 + 1)D, S3D, two-stream I3D, X3D, NL I3D). We can see that the accuracy

of our proposed Branch-Fusion Net was the highest on the small datasets of UCF-101 and HMDB51, reaching 97.32% (UCF-101) and 82.14% (HMDB51), which indicates that Branch-Fusion Net can fully understand the features on these two datasets. On the larger and more challenging Kinetics dataset, the Branch-Fusion Net also achieved state-of-the-art results, with 80.63% and 93.81% for TOP-1 and TOP-5, indicating that the Branch-Fusion Net can understand a large number of complex features.

4.6. Ablation Experiment of CSAM

The suppression of useless features and the highlighting of useful features will be reflected in the feature representation capability of the network with the CSAM added, and we chose the main 3D convolutional neural networks for the ablation experiment. The result is shown in Table 8.

Table 8. Accuracy comparison on the Kinetics dataset.

Architecture	Params/M	Accuracy (%)
C3D	58.378	72.93
C3D + CSAM	58.454	77.71
R3D	33.642	80.29
R3D + CSAM	33.731	83.90
R(2 + 1)D	33.641	83.08
R(2 + 1)D + CSAM	33.730	88.44
Branch-Fusion Net	14.856	89.63
Branch-Fusion Net + CSAM	16.190	93.55

The experimental result shows that the parameter overhead hardly increased after adding the CSAM to the networks, but there was a significant improvement in the accuracy rate, which indicates that the CSAM outperformed all the baselines without bells and whistles, demonstrating the general applicability of the CSAM across different architectures.

4.7. Comparison Experiment with Our Model

To evaluate the performance of our models (the Branch-Fusion Net with CSAM for the local feature extraction network and the Bi-GRU for the global feature extraction network), we conducted a comparison experiment with good feature extraction networks (P3D [43], R3D, and R (2 + 1)D) on the test set. The experimental result is shown in Figure 16.

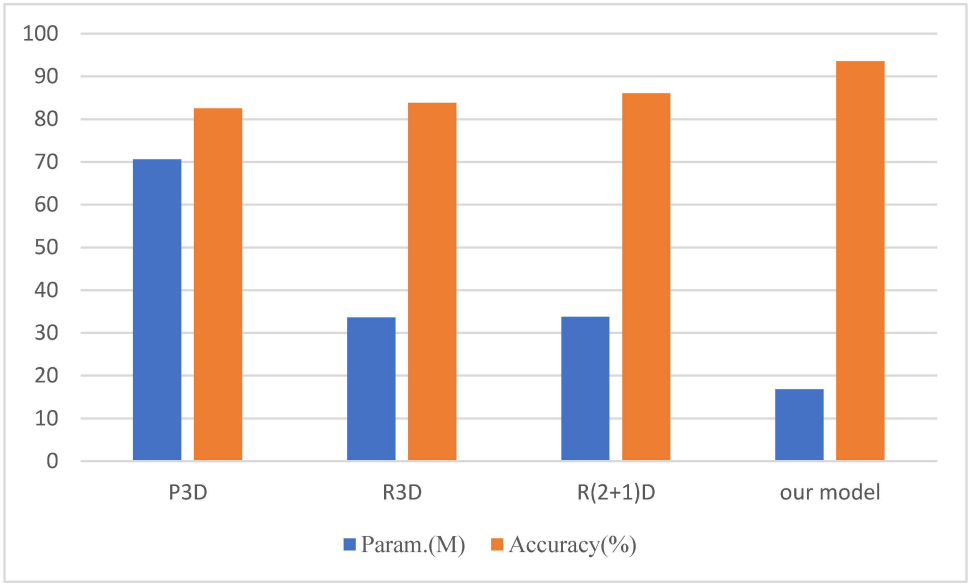

Figure 16. Comparison of parameter overhead and accuracy on the Crimes-mini dataset.

The experimental results show that our proposed model achieved the highest accuracy with the lowest parameter overhead. The low parameter overhead indicates that the model requires low memory, and the 93.55% accuracy indicates that our proposed model can adequately extract the key features in abnormal videos.

5. Conclusions

We proposed a model based on multiple-instance learning, which included the local feature extraction network and the global feature extraction network. For the local feature extraction network, we proposed the Branch-Fusion Net to reduce parameter overhead and improve the generalization ability at the same time. To prevent useless features from interfering with the model training, we proposed the CSAM to suppress useless features and enhance important features. For the global feature extraction network, we used a two-layer Bi-GRU to complete the global feature extraction. To make the model more suitable for mobile devices, we will reduce the parameter overhead using model compression techniques in the future.

Author Contributions: Data curation, P.Z.; software, Y.L. and P.Z.; supervision, Y.L.; visualization, P.Z.; writing—review and editing, P.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (61971007 and 61571013).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Nayak, R.; Pati, U.C.; Das, S.K. A comprehensive review on deep learning-based methods for video anomaly detection. *ScienceDirect* **2020**, *106*, 104078. [CrossRef]
2. Roshtkhari, M.J.; Levine, M.D. An on-line, real-time learning method for detecting anomalies in videos using spatio-temporal compositions. *Comput. Vis. Image Underst.* **2013**, *117*, 1436–1452. [CrossRef]
3. Li, Y.; Cai, Y.; Liu, J.; Lang, S.; Zhang, X. Spatio-Temporal Unity Networking for Video Anomaly Detection. *IEEE Access* **2019**, *7*, 172425–172432. [CrossRef]
4. Li, W.; Mahadevan, V.; Vasconcelos, N. Anomaly detection and localization in crowded scenes. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *36*, 18–32. [PubMed]
5. Xu, D.; Ricci, E.; Yan, Y.; Song, J.; Sebe, N. Learning deep representations of appearance and motion for anomalous event detection. *Comput. Vis. Image Underst.* **2017**, *156*, 117–127. [CrossRef]
6. Patraucean, V.; Handa, A.; Cipolla, R. Spatio-temporal video autoencoder with differentiable parameter. *Comput. Sci.* **2015**, *58*, 2415–2422.
7. Zhao, B.; Li, F.F.; Xing, E.P. Online detection of unusual events in videos via dynamic sparse coding. In Proceedings of the IEEE Conference on Computer Vision & Pattern Recognition, Colorado Springs, CO, USA, 20–25 June 2011.
8. Wang, X.Z.; Che, Z.P.; Jiang, B. Robust Unsupervised Video Anomaly Detection by Multipath Frame Prediction. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *33*, 2301–2312. [CrossRef]
9. Liu, W.; Luo, W.; Lian, D.; Gao, S. Future Frame Prediction for Anomaly Detection—A New Baseline. *arXiv* **2017**, arXiv:1712.09867.
10. Ionescu, R.T.; Khan, F.S.; Georgescu, M.I. Object-centric Auto-encoders and Dummy Anomalies for Abnormal Event Detection in Video. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019.
11. Kiran, B.; Dilip, T.; Ranjith, P. An overview of deep learning based methods for unsupervised and semi-supervised anomaly detection in videos. *J. Imaging* **2018**, *4*, 36. [CrossRef]
12. Tran, D.; Bourdev, L.; Fergus, R.; Torresani, L.; Paluri, M. Learning Spatiotemporal Features with 3D Convolutional Networks. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015.
13. Liang, Z.; Zhu, G.; Shen, P. Learning Spatiotemporal Features Using 3DCNN and Convolutional LSTM for Gesture Recognition. In Proceedings of the 2017 IEEE International Conference on Computer Vision Workshops (ICCVW), Venice, Italy, 22–29 October 2017.

14. Ouyang, X.; Xu, S.; Zhang, C.; Zhou, P.; Yang, Y.; Liu, G.; Li, X. A 3D-CNN and LSTM Based Multi-Task Learning Architecture for Action Recognition. *IEEE Access* **2017**, *7*, 40757–40770. [CrossRef]
15. Xu, X.; Liu, L.Q.; Zhang, L. Abnormal visual event detection based on multi instance learning and autoregressive integrated moving average model in edge-based Smart City surveillance. *Softw. Pract. Exp.* **2020**, *50*, 476–488. [CrossRef]
16. Jie, H.; Li, S.; Gang, S. Squeeze-and-Excitation Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *42*, 2021–2023.
17. Li, K.; Wang, Y.; Zhang, J. UniFormer: Unifying Convolution and Self-attention for Visual Recognition. *arXiv* **2022**, arXiv:2201.09450.
18. Hasan, M.; Choi, J.; Neumann, J. Learning Temporal Regularity in Video Sequences. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016.
19. Gong, D.; Liu, L.; Le, V. Memorizing Normality to Detect Anomaly: Parameter-Augmented Deep Autoencoder for Unsupervised Anomaly Detection. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019.
20. Park, H.; Noh, J.; Ham, B. Learning Parameter-guided Normality for Anomaly Detection. *arXiv* **2020**, arXiv:2003.13328.
21. Medel, J.R.; Savakis, A. Anomaly Detection in Video Using Predictive Convolutional Long Short-Term Parameter Networks. *arXiv* **2016**, arXiv:1612.00390.
22. Lu, Y.; Reddy, M.; Nabavi, S.S. Future Frame Prediction Using Convolutional VRNN for Anomaly Detection. In Proceedings of the 2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Taipei, Taiwan, 18–21 September 2019.
23. Mathieu, M.; Couprie, C.; Lecun, Y. Deep multi-scale video prediction beyond mean square error. *arXiv* **2015**, arXiv:1511.05440.
24. Ye, M.; Peng, X.; Gan, W. AnoPCN: Video Anomaly Detection via Deep Predictive Coding Network. In Proceedings of the 27th ACM International Conference on Multimedia, Nice, France, 21–25 October 2019.
25. Sabokrou, M.; Khalooei, M.; Fathy, M.; Adeli, E. Adversarially Learned One-Class Classifier for Novelty Detection. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018.
26. Wu, P.; Liu, J.; Shen, F. A Deep One-Class Neural Network for Anomalous Event Detection in Complex Scenes. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *31*, 2609–2622. [CrossRef] [PubMed]
27. Xu, K.; Sun, T.; Jiang, X. Video Anomaly Detection and Localization Based on an Adaptive Intra-frame Classification Network. *IEEE Trans. Multimed.* **2020**, *22*, 394–406. [CrossRef]
28. Sultani, W.; Chen, C.; Shah, M. Real-World Anomaly Detection in Surveillance Videos. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018.
29. Kamoona, A.M.; Gosta, A.K.; Bab-Hadiashar, A. Multiple Instance-Based Video Anomaly Detection using Deep Temporal Encoding-Decoding. *arXiv* **2020**, arXiv:2007.01548. [CrossRef]
30. Zhu, Y.; Newsam, S. Motion-Aware Feature for Improved Video Anomaly Detection. *arXiv* **2019**, arXiv:1907.10211.
31. Simonyan, K.; Zisserman, A. Two-Stream Convolutional Networks for Action Recognition in Videos. In Proceedings of the 27th International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 8 December 2014.
32. Carreira, J.; Zisserman, A. Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
33. Wang, L.; Xiong, Y.; Wang, Z.; Qiao, Y.; Lin, D.; Tang, X.; Van Gool, L. Temporal Segment Networks: Towards Good Practices for Deep Action Recognition. *arXiv* **2016**, arXiv:1608.00859.
34. Lin, J.; Gan, C.; Han, S. TSM: Temporal Shift Module for Efficient Video Understanding. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019.
35. Li, Y.; Ji, B.; Shi, X.; Zhang, J.; Kang, B.; Wang, L. TEA: Temporal Excitation and Aggregation for Action Recognition. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020.
36. Wang, L.; Tong, Z.; Ji, B.; Wu, G. TDN: Temporal Difference Networks for Efficient Action Recognition. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021.
37. Feichtenhofer, C.; Fan, H.; Malik, J.; He, K. SlowFast Networks for Video Recognition. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019.
38. Hara, K.; Kataoka, H.; Satoh, Y. Learning Spatio-Temporal Features with 3D Residual Networks for Action Recognition. In Proceedings of the 2017 IEEE International Conference on Computer Vision Workshops (ICCVW), Venice, Italy, 22–29 October 2017.
39. Tran, D.; Wang, H.; Torresani, L.; Ray, J.; LeCun, Y.; Paluri, M. A Closer Look at Spatiotemporal Convolutions for Action Recognition. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018.
40. Xie, S.; Sun, C.; Huang, J.; Tu, Z.; Murphy, K. Rethinking Spatiotemporal Feature Learning: Speed-Accuracy Trade-offs in Video Classification. *arXiv* **2017**, arXiv:1712.04851.
41. Feichtenhofer, C. X3D: Expanding Architectures for Efficient Video Recognition. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020.

42. Wang, X.; Girshick, R.; Gupta, A.; He, K. Non-Local Neural Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018.
43. Qiu, Z.; Yao, T.; Mei, T. Learning Spatio-Temporal Representation with Pseudo-3D Residual Networks. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017.