

Article

Not peer-reviewed version

---

# DualCascadeTSF-MobileNetV2: A Lightweight Violence Behavior Recognition Model

---

[Yuang Chen](#) , [Yong Li](#) <sup>\*</sup> , Shaohua Li , Shuhan Lv , Fang Lin

Posted Date: 5 March 2025

doi: 10.20944/preprints202503.0316.v1

Keywords: deep learning; lightweight models; violence behavior recognition; edge devices




Preprints.org is a free multidisciplinary platform providing preprint service that is dedicated to making early versions of research outputs permanently available and citable. Preprints posted at Preprints.org appear in Web of Science, Crossref, Google Scholar, Scilit, Europe PMC.

Copyright: This open access article is published under a Creative Commons CC BY 4.0 license, which permit the free download, distribution, and reuse, provided that the author and preprint are cited in any reuse.

## Article

# DualCascadeTSF-MobileNetV2: A Lightweight Violence Behavior Recognition Model

Yuang Chen <sup>1,2</sup> , Yong Li <sup>1,\*</sup>, Shaohua Li <sup>1</sup>, Shuhan Lv <sup>1,2</sup> and Fang Lin <sup>1,2</sup>

<sup>1</sup> Engineering University of PAP, Key laboratory of CTC & IE(Engineering University of PAP),Ministry of Education, Xi'an 710086, China

<sup>2</sup> Engineering University of PAP, Graduate Student Brigade, Xi'an 710086, China

\* Correspondence: liyong@nudt.edu.cn

**Abstract:** This paper proposes a lightweight violent behavior recognition model, DualCascadeTSF-MobileNetV2, which is improved based on the temporal shift module and its subsequent research. By introducing the Dual Cascade Temporal Shift and Fusion module, the model further enhances the feature correlation ability in the time dimension and solves the problem of information sparsity caused by multiple temporal shifts. Meanwhile, the model incorporates the efficient lightweight structure of MobileNetV2, significantly reducing the number of parameters and computational complexity. Experiments were conducted on three public violent behavior datasets, Crowd Violence, RWF-2000, and Hockey Fights, to verify the performance of the model. The results show that it outperforms other classic models in terms of accuracy, computational speed, and memory size, especially among lightweight models. This research continues and expands on the previous achievements in the fields of TSM and lightweight network design, providing a new solution for real-time violent behavior recognition on edge devices.

**Keywords:** deep learning; lightweight models; violence behavior recognition; edge devices

## 1. Introduction

Social safety and stability are the desired living conditions for every citizen. To prevent and reduce the occurrence of violent behavior, a large number of security personnel monitor social security conditions in real-time through surveillance cameras or frontline duty terminal devices. This monitoring method has significantly improved the response speed to violent behavior. However, relying solely on human supervision has two main issues: on the one hand, it is difficult for surveillance personnel to maintain high concentration at all times, leading to potential misjudgments or missed detections of violent behavior; on the other hand, with the vast amount of surveillance data, it is difficult for personnel to quickly analyze complex situations and make effective judgments. Therefore, introducing intelligent monitoring systems to assist security personnel in detecting violent behavior is of great importance.

The core technology of intelligent monitoring systems lies in intelligent recognition of violent behavior, which is an important direction in the field of computer vision for behavior recognition. In recent years, deep learning-based behavior recognition methods have made significant progress, with many new models achieving increasingly precise results on public datasets. However, considering the practical needs of violent behavior recognition tasks, models need not only higher accuracy but also faster recognition speed to reduce latency, fewer parameters, and smaller model size for deployment on edge devices.

In this context, we noticed the temporal shift module (TSM) designed by Lin et al. [1]. This model interacts with the action information at different times by shifting the time steps of different channels in the temporal dimension. This allows the extraction of action features across three consecutive time steps with only a single 2D convolution, effectively achieving 3D convolutional performance with 2D

convolutional efficiency without sacrificing accuracy. This significantly improves computational speed and reduces recognition latency.

The design of TSM is very ingenious and still has potential for further exploration. For example, a longer-term information correlation can be achieved by fusing the features before and after the shift at the same time point; the number of parameters and the model volume can be further reduced by replacing the base model. Based on TSM and its subsequent research, this paper deeply explores its potential optimization space. The main contributions are as follows:

1. Improved TSM based on previous studies and designed the Dual Cascade Temporal Shift and Fusion (DualCascadeTSF) module. This module integrates the feature fusion method into TSM, strengthening the connection in the temporal dimension between adjacent time steps without increasing the number of parameters.
2. Combined the DualCascadeTSF module with the MobileNetV2. This not only improves the accuracy of MobileNetV2, but also makes it have fewer parameters and a faster operation speed compared to other classic models, making it more suitable for deployment on edge devices.

## 2. Related Work

### 2.1. Research on Lightweight Models

In the current development stage of behavior recognition tasks, researchers' attention is no longer limited to continuously improving the recognition accuracy of models. Instead, they pay more attention to lightweight models with fewer parameters, a smaller size, and shorter inference times. Such models are crucial for effective deployment in environments with limited hardware resources, especially in scenarios that require edge computing, such as mobile devices, embedded systems, and miniaturized machines.

Currently, there are three main methods to obtain lightweight models: the first is to prune classic models; the second is to design lightweight models based on manual experience; and the third is to use the Neural Architecture Search (NAS) to generate efficient models.

Pruning is one of the most commonly used methods to obtain lightweight models. This method reduces the size and computational complexity of classic models by pruning weights, channels, or structures while maintaining high performance. VGGNet pruning [2] achieves channel pruning by applying L1 regularization to the scaling factor of the Batch Normalization (BN) layer during the network training process, which is widely used in image classification and object detection tasks. Inception pruning [3] evaluates the importance of each convolutional filter and removes less significant filters to reduce the size of the model and accelerate inference. BERT pruning [4] significantly reduces the number of model parameters and computational costs by removing redundant attention heads and layers, which is applicable to natural language processing tasks.

The design of lightweight models based on human experience is a traditional but still effective method for lightweight model. This method relies on experts' knowledge and experience to design network structures and optimize components to achieve model lightweighting. The main methods include using lightweight convolutions, designing compact network structures, using lightweight activation functions, and optimizing pooling and normalization. MobileNetV1 [5] introduces depthwise separable convolutions, decomposing standard convolutions into depthwise and pointwise convolutions, significantly reducing computational load while using ReLU6 activation function to limit output range and avoid numerical overflow. MobileNetV2 [6] further improves this by adopting inverted residual blocks and linear bottlenecks, expanding channels through expansion layers, processing feature maps with depthwise separable and pointwise convolutions, and then compressing back to the input channel number. ShuffleNet [7] uses group convolutions and channel shuffle techniques to enhance the information exchange between feature maps while reducing the amount of calculation, and combines residual connections to improve model performance. SqueezeNet [8] significantly reduces the number of parameters through the fire module (including the squeeze layer and the expand layer) and global average pooling.

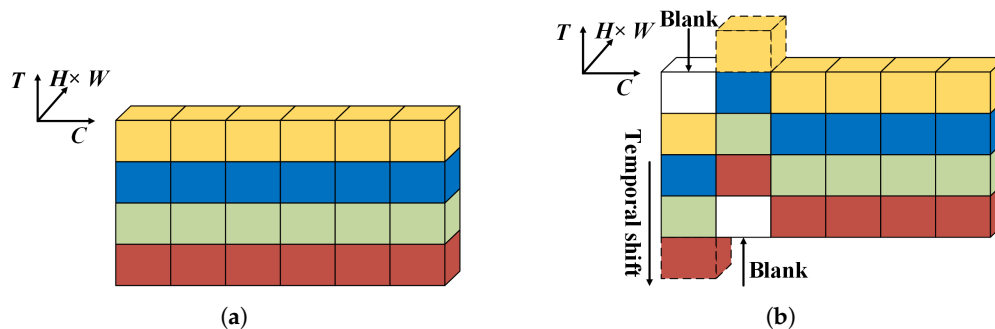
NAS is a model lightweighting method that automatically searches for the optimal neural network structure. Its implementation steps include setting goals, defining the search space, determining the search strategy, and evaluating performance. Google designed MobileNetV3 [9] through the NAS method based on reinforcement learning, optimizing the network structure, which is applicable to image classification, object detection, and real-time video processing tasks. EfficientNet-Lite [10], as a lightweight version of EfficientNet, further optimized the model structure through the NAS method and is widely used in image classification, object detection, and semantic segmentation tasks. BiX-Net [11] used a two-stage NAS algorithm to automatically search for an ultra-lightweight and effective bidirectional skip connection structure, suitable for medical image segmentation tasks. LightTrack [12] optimized the network structure for object tracking tasks through the NAS method and is applicable to object tracking and video analysis tasks. OFA (Once-for-All) [13] generates a supernet through a one-time NAS search, from which multiple subnetworks are pruned, each optimized for different tasks and resource constraints, suitable for image classification, object detection, and semantic segmentation tasks.

## 2.2. TSM and Two-Cascade TSM

In behavior recognition tasks, different from static image input, the number of video streams parameters shows explosive growth, which poses a huge challenge to achieving high-accuracy and low-cost behavior recognition.

Traditional 2D convolutional neural networks (CNNs) have a low computational cost but cannot capture the unique temporal information of videos; while 3D CNNs can achieve good performance, but their computational load is huge and the deployment cost is too high. To solve this problem, the Temporal Shift Module (TSM) came into being, which cleverly achieved the goal of approaching the performance of 3D CNNs with the computational cost of 2D CNNs.

Specifically, the working principle of TSM is shown in Figure 1. Figure 1(a) represents the video stream before temporal shift, where  $C$  is the channel dimension, representing different channels of the video stream, and  $T$  is the temporal dimension, with each layer of color corresponding to a time step in the sequence. Figure 1(b) shows the video stream after temporal shifting, where some channels are shifted upward or downward by one time step in the temporal dimension, while the remaining parts stay unchanged. Through this approach, the task of extracting behavior features across any three consecutive time steps can be accomplished with a single convolution operation. meaning that 2D convolution alone can be used to extract spatio-temporal features for actions at a single time step.



**Figure 1.** The process of the video stream going through the Temporal Shift Module operation. (a) The video stream before temporal shifting and (b) The video stream after temporal shifting.

Therefore, the operation of TSM can be divided into two steps: temporal shifting and convolution. Suppose that there is a  $1 \times 3$  convolution kernel  $W = (\omega_1, \omega_2, \omega_3)$  and an infinitely long video stream  $X_{n \times 3}^k = (X_{i-k,j}, X_{i,j}, X_{i+k,j}) \in \mathbb{R}^{n \times 3}$  with 3 channels, where  $k$  represents the number of temporal shifts.

After one temporal shift, the video stream becomes  $X_{n \times 3}^1 = (X_{i-1,j}, X_{i2}, X_{i+1,j}) \in \mathbb{R}^{n \times 3}$ , and then the convolution operation is performed:

$$y_i = \omega_1 x_{i-1,1} + \omega_2 x_{i,2} + \omega_3 x_{i+1,3} \quad (1)$$

$$Y = \sum_{j=1}^3 (\omega_j x_{i-1,j}) \quad (2)$$

Where  $y_i$  represents the convolution operation process, and  $Y$  represents the calculation result. From the calculation process, it can be seen that the first step of the TSM operation does not generate computational costs, while the second step will generate a certain amount of calculation. However, compared to the reduction in computational cost from transforming the convolution operation from 3D to 2D, the slight increase in computation is acceptable.

Based on this, Liang et al.[14] further proposed the Two-cascade TSM. This module enhances the temporal correlation ability of the model by cascading another TSM within the residual block, enabling it to capture longer-term information dependencies. This improvement significantly boosts the model's performance and achieves more excellent results.

### 2.3. MobileNetV2

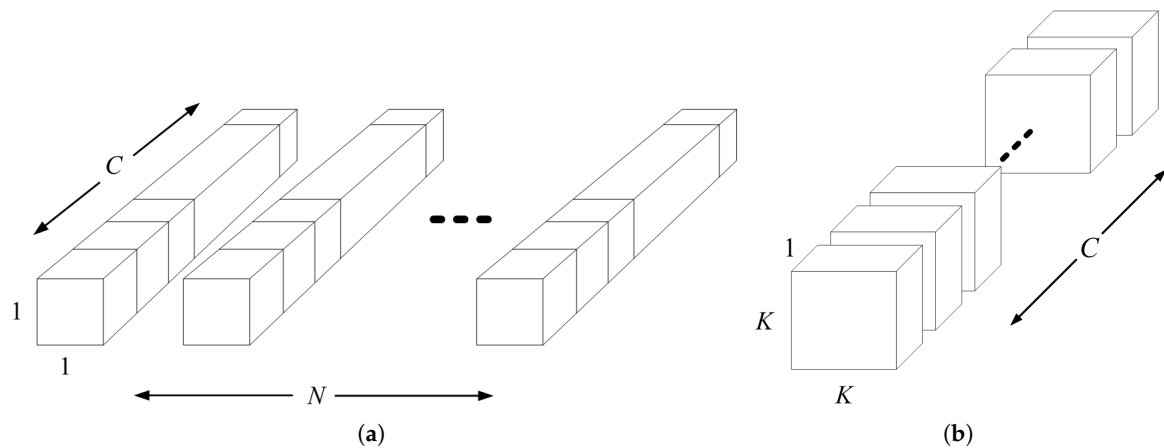
MobileNetV2 is a lightweight convolutional neural network developed based on MobileNetV1, both of which aim to provide efficient computing capabilities for mobile devices and resource-constrained environments. MobileNetV2 inherits the core idea of MobileNetV1, that is, replacing traditional standard convolutions with depthwise separable convolutions to significantly reduce the amount of calculation and the number of parameters.

As shown in Figure 2, depthwise separable convolution divides standard convolution into pointwise convolution and depthwise convolution. The main function of depthwise convolution is to extract features from each input channel using a single filter, while pointwise convolution applies a  $1 \times 1$  convolution to combine the output of depthwise convolution. The computational cost generated by this method is  $K \times K \times C \times H \times W + C \times N \times H \times W$ , where  $K \times K$  represents the size of the convolution kernel,  $C$  is the number of channels,  $H \times W$  is the size of the input video frame, and  $N$  is the number of input video frames. Compared to the computational cost  $K \times K \times H \times W \times C \times N$  generated by standard convolution, which extracts and combines features simultaneously, the reduced computational cost is:

$$\frac{K \times K \times C \times H \times W + C \times N \times H \times W}{K \times K \times H \times W \times C \times N} = \frac{1}{N} + \frac{1}{K^2} \quad (3)$$

It can be seen that this algorithm indeed achieves the expected effect of reducing the amount of calculation and the number of parameters.





**Figure 2.** The two convolution forms of depthwise separable convolution. (a) Pointwise Convolution and (b) Depthwise Convolution.

Meanwhile, MobileNetV2 introduces two important innovations based on MobileNetV1: Inverted Residuals and Linear Bottlenecks. The core idea of the linear bottleneck is to avoid using nonlinear activation functions in the low-dimensional bottleneck layer to ensure the integrity of the input information and prevent the loss of key information in the video frames. In contrast, non-linear activation functions are introduced in the high-dimensional expansion layer to enhance the model's expressive power. The design of the Inverted Residuals is opposite to that of the traditional residual block. The shortcut connections are located between the low-dimensional bottleneck layers, while the intermediate layers perform feature extraction and nonlinear transformations using lightweight depthwise separable convolutions. This design not only significantly reduces the number of model parameters, but also further improves computational efficiency, making MobileNetV2 more suitable for deployment and operation in resource-constrained environments.

### 3. Methods

#### 3.1. Conception

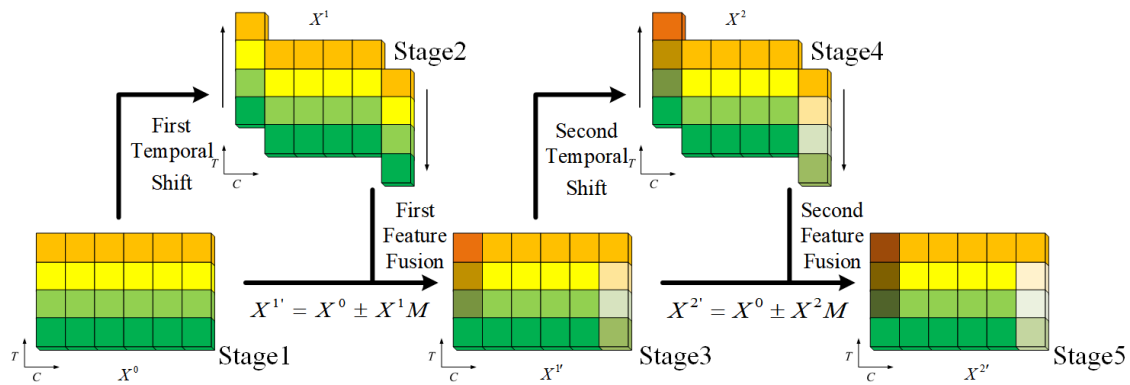
The Two-cascade TSM proposed in the literature [14] is an improvement on the traditional TSM, which significantly enhances the model's ability to extract long-term information. However, when attempting to cascade more TSMs within the same residual block, the experimental results did not meet expectations. The authors speculate that this might be due to the excessive temporal shifting operations, which cause the originally adjacent feature maps in different channels to become sparse in the temporal dimension, thereby weakening the model's performance.

Based on this observation, this paper proposes a new hypothesis: if a feature fusion step is added after each temporal shift to combine the feature maps at the same temporal position before and after the shift, could this further enhance the model's ability to extract longer-term temporal information? Through feature fusion, the associations between features in the temporal dimension can be strengthened without adding additional parameters, thereby compensating for the information sparsity caused by multiple shifting operations.

Moreover, to make the designed model more lightweight, this paper chooses to replace traditional ResNet with MobileNetV2 as the base model. MobileNetV2, with its unique inverted residual structure and linear bottleneck design, significantly reduces the number of parameters and computational complexity while maintaining performance. Therefore, combining the DualCascadeTSF module with MobileNetV2 not only can achieve higher accuracy but also effectively reduce the model size and improve the calculation speed, making it more suitable for deployment on resource-constrained edge devices.

### 3.2. DualCascadeTSF Module

Based on the concept mentioned above, this paper proposes a module, the Dual Cascade Temporal Shift and Fusion (DualCascadeTSF) module. The design of this module inherits the core idea of the TSM, which involves shifting the temporal positions of video frame channels to alter the manner of convolution operations, thereby improving computational efficiency. Building on this foundation, the DualCascadeTSF module further introduces a feature fusion mechanism that merges feature maps at the same temporal position during each temporal shift. This enhances the model's ability to extract longer-term information. As shown in Figure 3, the working principle of the DualCascadeTSF module is illustrated. Stage 1 represents the original input video stream, where each column corresponds to a channel and each row represents a frame. In this figure, we assume that it has 4 frames and 6 channels. Stage 2 involves the first temporal shift, where one channel is shifted forward in time, while another channel is shifted backward. Stage 3 shows the feature fusion of the original input video stream with the shifted video stream after the first temporal shift. Stage 4 involves a second temporal shift after the first feature fusion. Stage 5 shows the feature fusion of the original input video stream with the shifted video stream after the second temporal shift. At this point, the DualCascadeTSF module completes its entire operation. It can be seen from the figure that the working principle of the DualCascadeTSF module is very simple and does not introduce significant additional parameters.



**Figure 3.** The working principle of the DualCascadeTSF module. Stage 1 represents the original state of the input video stream, Stage 2 involves the first temporal shift of the video stream, Stage 3 is the feature fusion of the video stream after the first temporal shift, Stage 4 involves the second temporal shift of the video stream, Stage 5 is the feature fusion of the video stream after the second temporal shift.

After intuitively presenting the working principle of the module in the form of diagrams, this paper further explains the ingenuity of the module's design from a mathematical perspective. Suppose that there are a  $1 \times 3$  convolution kernel  $W = (\omega_1, \omega_2, \omega_3)$  and a 3-channel infinitely long video stream  $X_{n \times 3}^k = (X_{i-k,j}, X_{i,j}, X_{i+k,j}) \in \mathbb{R}^{n \times 3}, (n = \infty)$ , where  $k$  represents the number of temporal shifts,  $i$  represents the position of the time step in the video stream,  $j$  represents the position of the channel in the video stream, and  $n$  represents the number of frames in the input video stream. After one temporal shift, the video stream becomes  $X_{n \times 3}^1 = (X_{i-1,j}, X_{i,j}, X_{i+1,j}) \in \mathbb{R}^{n \times 3}, (n = \infty)$ , and then the convolution operation is performed:

$$y_i^1 = \omega_1 x_{i-1,1} + \omega_2 x_{i,2} + \omega_3 x_{i+1,3} \quad (4)$$

$$Y^1 = \sum_{j=1}^3 (\omega_j x_{i-1,j}, \omega_j x_{i,j}, \omega_j x_{i+1,j}) \quad (5)$$

Where  $y_i^1$  represents the process of the first convolution operation, and  $Y^1$  represents the calculation result. After the first temporal shift, feature fusion is performed:

$$X^{1'} = X^0 \pm X^1 M \quad (6)$$

Where  $X^{1'}$  represents the feature values of the infinitely long video stream after the first feature fusion,  $M$  is the  $\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$  matrix, which is used to extract the columns that have undergone temporal shifting in  $X^1$ , namely the feature values in columns 1 and 3. At this point,  $X^{1'}$  is:

$$X^{1'} = (x_{i,1} \pm \omega_1 x_{i-1,1}, x_{i,2}, x_{i,3} \pm \omega_3 x_{i-1,3})_{n \times 3} \quad (7)$$

After completing the first temporal shift and feature fusion, the second temporal shift is then performed:

$$y_i^2 = \omega_1(x_{i-1,1} \pm \omega_1 x_{i-2,1}) + \omega_2 x_{i,2} + \omega_3(x_{i+1,3} \pm \omega_3 x_{i+2,3}) \quad (8)$$

$$Y^2 = \sum_{j=1}^3 [\omega_j(x_{i-1,j} \pm \omega_j x_{i-2,j}) + \omega_j x_{i,j} + \omega_j(x_{i+1,j} \pm \omega_j x_{i+2,j})] \quad (9)$$

Here,  $y_i^2$  represents the process of the first convolution operation, and  $Y^2$  represents the calculation result. After the second temporal shift, feature fusion is performed:

$$X^{2'} = X^0 \pm X^2 M \quad (10)$$

Where  $X^{2'}$  represents the feature values of the infinitely long video stream after the second feature fusion, and at this point, its value is:

$$X^{2'} = [x_{i,1} \pm \omega_1(x_{i,1} \pm \omega_1 x_{i-1,1}), x_{i,2}, x_{i,3} \pm \omega_3(x_{i,3} \pm \omega_3 x_{i+1,3})]_{n \times 3} \quad (11)$$

After simplification, it is:

$$X^{2'} = (x_{i,1} \pm \omega_1 x_{i,1} \pm \omega_1^2 x_{i-1,1}, x_{i,2}, x_{i,3} \pm \omega_3 x_{i,3} \pm \omega_3^2 x_{i+1,3})_{n \times 3} \quad (12)$$

At this point, it can be clearly seen that after two rounds of temporal shift and feature fusion, each shifted channel in every time step contains the features before the shift. When the model starts the convolution operation, each time step for convolution contains not only the original feature values of the unshifted channels (such as the second column in Eq. (12)) but also the feature values of the channels shifted forward and backward in the temporal dimension (such as the first and third columns in Eq. (12)). By this method, the correlation between feature maps and between channels can be improved, thus greatly enhancing the robustness of the model.

### 3.3. DualCascadeTSF-MobileNetV2

To meet the demand for lightweight and efficient algorithms for intelligent violence recognition on edge devices, this paper proposes a new model—DualCascadeTSF-MobileNetV2—by combining DualCascadeTSF module with MobileNetV2. DualCascadeTSF module enhance the ability to extract temporal features through temporal shifting, while MobileNetV2 is renowned for its efficient lightweight structure. The combination of the two provides an effective solution for achieving a high-performance model with low computational requirements.

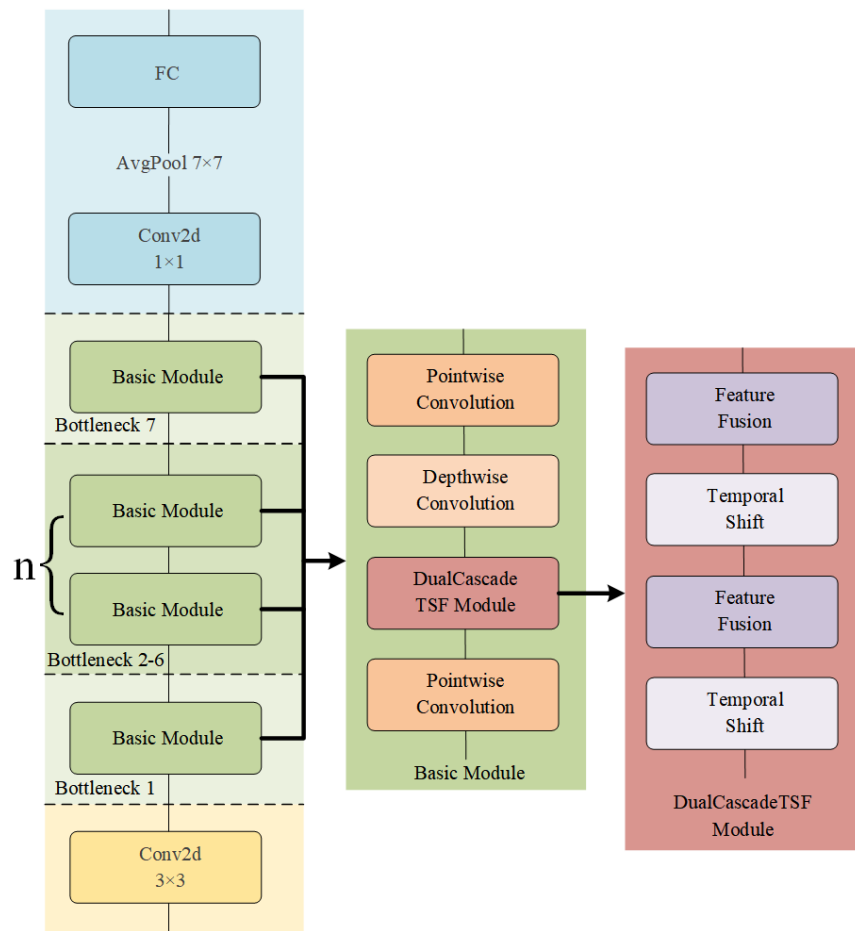
Table 1 briefly lists the core structural design of DualCascadeTSF-MobileNetV2. Here, “conv” denotes the convolutional layer, “bottleneck” denotes the bottleneck layer, and “Avgpool” denotes the average pooling layer.



Table 1. Structure of DualCascadeTSF-MobileNetV2.

Layer	Structure	DualCascadeTSF Included	Output Size	Repetition Times
conv1	$3 \times 3 \times 32$		$112 \times 112$	$\times 1$
bottleneck1	$1 \times 1 \times 32, 3 \times 3 \times 32, 1 \times 1 \times 16$	Yes	$112 \times 112$	$\times 1$
bottleneck2	$1 \times 1 \times 96, 3 \times 3 \times 96, 1 \times 1 \times 24$	Yes	$56 \times 56$	$\times 2$
bottleneck3	$1 \times 1 \times 192, 3 \times 3 \times 192, 1 \times 1 \times 32$	Yes	$28 \times 28$	$\times 3$
bottleneck4	$1 \times 1 \times 384, 3 \times 3 \times 384, 1 \times 1 \times 64$	Yes	$14 \times 14$	$\times 4$
bottleneck5	$1 \times 1 \times 576, 3 \times 3 \times 576, 1 \times 1 \times 96$	Yes	$14 \times 14$	$\times 3$
bottleneck6	$1 \times 1 \times 960, 3 \times 3 \times 960, 1 \times 1 \times 160$	Yes	$7 \times 7$	$\times 3$
bottleneck7	$1 \times 1 \times 960, 3 \times 3 \times 960, 1 \times 1 \times 320$	Yes	$7 \times 7$	$\times 1$
conv2	$1 \times 1 \times 1280$		$7 \times 7$	$\times 1$
Avgpool	Pool $7 \times 7$		$1 \times 1$	$\times 1$
FC	Linear $1280 \rightarrow 2$		$1 \times 1$	$\times 1$

As shown in Figure 4, it is the overall structure of DualCascadeTSF-MobileNetV2, which displays the specific deployment positions of the DualCascadeTSF module. From the figure, it can be seen that in this paper, the DualCascadeTSF module is designed to be placed before each depthwise convolution. The main reason is that the first pointwise convolution in the bottleneck layer usually increases the number of channels of the input feature map, making the information of the feature map richer. In this case, temporal features can be extracted more effectively through temporal shift operations. In addition, inserting the DualCascadeTSF module at this position not only avoids the loss of information caused by too few channels but also prevents the problem of increasing the computational burden due to too many channels, thus achieving a balance between performance and efficiency.



**Figure 4.** Overall Architecture of DualCascadeTSF-MobileNetV2. The left part is the entire structure of DualCascadeTSF-MobileNetV2. The middle part is the structure of one bottleneck in DualCascadeTSF-MobileNetV2. The right part is the structure of one DualCascadeTSF within the bottleneck.

## 4. Experiments

### 4.1. Datasets

To verify the performance of the DualCascadeTSF-MobileNetV2 model in violence recognition tasks, this paper selected three publicly available violence-related datasets — Crowd Violence [15], RWF-2000 [16], Hockey Fights [17] — for experimentation.

The Crowd Violence dataset, also known as the Violent Flow dataset, mainly focuses on scenes of crowd violence. This dataset contains 246 video clips, with the video lengths ranging from 1 second to 6.5 seconds, and the average duration is approximately 3.6 seconds. Due to the low performance of the shooting equipment and the large scene, the overall video images are rather blurry. However, these videos cover a variety of violent behaviors, such as fights and brawls. And in order to provide a comparative analysis, the dataset also includes some video clips of non-violent behaviors, which are used to enhance the model's ability to distinguish between different behaviors.



**Figure 5.** The Crowd Violence dataset.

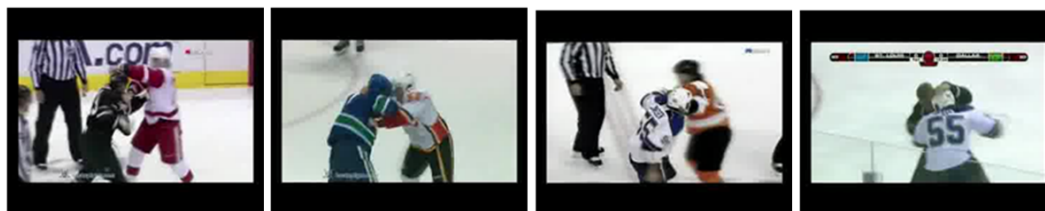
The RWF-2000 dataset is a newly proposed medium-sized violent behavior dataset, which contains 2000 surveillance video clips collected from YouTube. Among them, the training set includes 1600

video clips, and the validation set includes 400 video clips. Each video clip is 5 seconds long, with a total of 150 frames, and has not undergone any preprocessing. This dataset demonstrates violent behaviors between two people, multiple people, and crowds. The scenes are complex and diverse, and the recognition difficulty is relatively high, which is very close to the actual application requirements and has a high research value. In addition, the dataset not only contains videos of violent behaviors but also videos of nonviolent behaviors as a control group to help the model better learn the differences between the two types of behavior.



**Figure 6.** The RWF-2000 dataset.

The Hockey Fights dataset contains 1000 video clips collected from ice hockey games. Among them, the training set has 800 video clips and the validation set has 200 video clips. Each clip is 2 seconds long and contains 41 frames. This dataset records the fighting behaviors of ice hockey players during the games, as well as the physical contacts in normal games as a control group. This dataset is not only of great significance for the safety management of sports events but also provides valuable resources for the research in fields such as human behavior recognition and abnormal behavior detection.

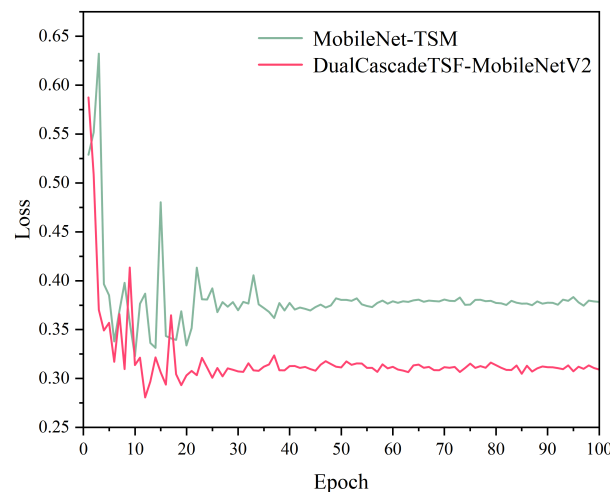


**Figure 7.** The Hockey Fights dataset.

#### 4.2. Parameter Settings

The environmental configuration used in the experiment is as follows: The operating system is Ubuntu 16.04, the deep learning framework is PyTorch 1.9.0, the CPU is Intel I9-10920X. At the same time, CUDA 11.1 is utilized to accelerate GPU computing, and two NVIDIA RTX 2080 Ti GPUs (with 11 GB of video memory per card) are equipped for parallel computing. During the experiment, the base model MobileNetV2 is initialized using the pre-trained parameters provided by PyTorch, and the training and testing are completed based on this. The Stochastic Gradient Descent (SGD) algorithm is selected as the optimization algorithm. The initial learning rate is set to 0.000375, and the learning rate adjustment strategy is to decay the learning rate to 80% of its original value every 2 epochs. The input modality is RGB images, the batch size is set to 8, and the entire training process runs for 100 epochs.

In order to verify whether the model designed in this paper will exhibit overfitting, we replicated the experiment in literature [18] on the RWF-2000 dataset and compared the loss curve of the model in this paper with the results of the replicated experiment. The experimental results are shown in Figure 8. The light red line and the light green line represent the loss value change curves of DualCascadeTSF-MobileNetV2 and MobileNet-TSM, respectively. As can be seen from the figure, after 20 epochs, the loss values of the two curves gradually converge, and the loss value of DualCascadeTSF-MobileNetV2 is significantly lower than that of MobileNet-TSM. This result indicates that the model proposed in this paper not only has stronger generalization ability but also does not have the problem of overfitting.



**Figure 8.** Loss Value Curves of the Two Models.

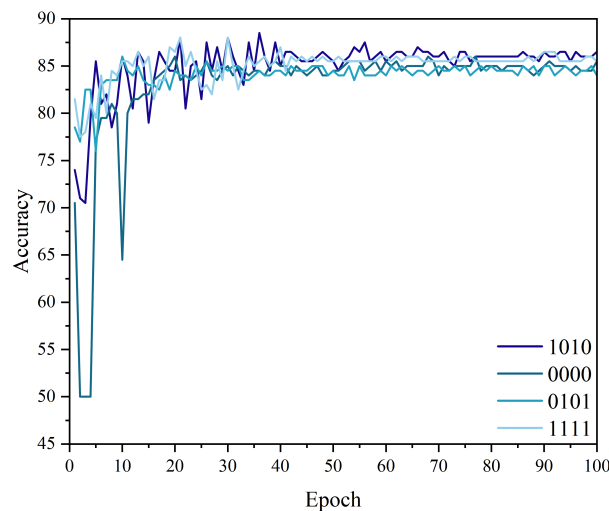
### 4.3. Results

#### 4.3.1. Ablation Experiments of Different Feature Fusion Methods

Firstly, this paper conducts an ablation experiment on the feature fusion methods, focusing on examining the impacts of three key factors on the model's performance: the batch of temporal shift (the first shift, the second shift), the direction of temporal shift (forward, backward), and the feature fusion algorithm (addition or subtraction). By testing different combinations in the experiment, the aim is to determine the optimal feature fusion strategy, so as to improve the overall performance of the model.

Since the naming of the feature fusion methods involved in the experiment is rather complicated, for the convenience of description, we use serial numbers to represent different feature fusion methods. The specific rules are as follows: From left to right, the first digit represents the operation mode of the first forward shift, the second digit represents the operation mode of the first backward shift, the third digit represents the operation mode of the second forward shift, and the fourth digit represents the operation mode of the second backward shift. Among them, "1" represents the addition operation, and "0" represents the subtraction operation. For example, the combination mode of "using the addition operation for the first forward shift, using the subtraction operation for the first backward shift, using the addition operation for the second forward shift, and using the subtraction operation for the second backward shift" corresponds to the serial number "1010".

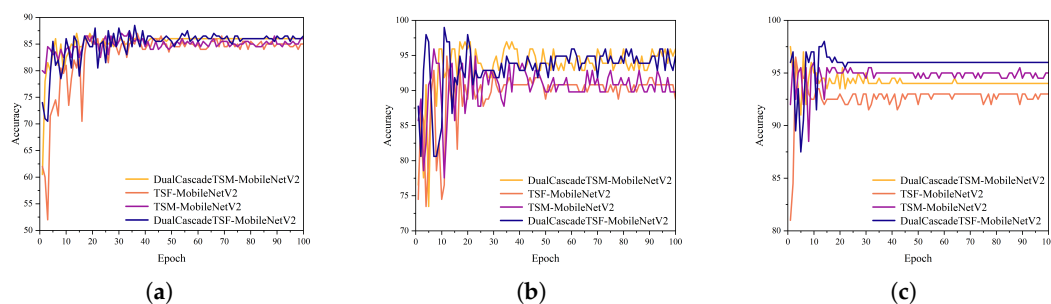
After 100 epochs of training and validation on the RWF-2000 dataset, the experimental results are shown in Figure 9. The figure displays the accuracy curves of four different feature fusion methods, corresponding to the codes 1010, 0000, 0101, and 1111, represented by blue lines with shades darkening in the order of the codes. It can be seen from the figure that the method with code 1010 (dark blue line), which represents the feature fusion strategy of "using addition for the first forward shift, subtraction for the first backward shift, addition for the second forward shift, and subtraction for the second backward shift," has an accuracy curve that is significantly higher than the other methods. It also demonstrates better convergence and stability with minimal fluctuations. Therefore, this feature fusion method has been experimentally validated as the optimal solution, enabling the model to achieve the best performance. This method will be adopted for feature fusion between feature maps in subsequent experiments.



**Figure 9.** Accuracy curves of four different feature fusion methods on the RWF-2000 dataset.

#### 4.3.2. Ablation Study on Different Structures

To verify whether the proposed model suffers from performance degradation due to structural complexity, we simplified the model structure and compared the performance of different modules combined with MobileNetV2. Four module combinations were selected for the experiments: DualCascadeTSM-MobileNetV2, TSF-MobileNetV2, TSM-MobileNetV2, and DualCascadeTSF-MobileNetV2. These combinations were evaluated on three datasets: Crowd Violence, RWF-2000, and Hockey Fights. After 100 epochs of training, the experimental results are shown in Figure 10



**Figure 10.** Experimental accuracy curves: (a) Accuracy curve on the Crowd Violence dataset; (b) Accuracy curve on the RWF-2000 dataset; (c) Accuracy curve on the Hockey Fights dataset.

Figure 10(a) shows the accuracy curve of the model on the Crowd Violence dataset. It can be seen from the figure that the accuracy of DualCascadeTSF-MobileNetV2 and DualCascadeTSM-MobileNetV2 is significantly higher than that of TSF-MobileNetV2 and TSM-MobileNetV2. Figure 10(b) shows the accuracy curve of the model on the RWF-2000 dataset. It can be seen that the accuracy of DualCascadeTSF-MobileNetV2 is significantly better than the other three models. Figure 10(c) shows the accuracy curve of the model in the Hockey Fights dataset. The results indicate that DualCascadeTSF-MobileNetV2 not only achieves the highest accuracy but also maintains stability after convergence.

The ablation experiments demonstrate that the proposed DualCascadeTSF-MobileNetV2 does not suffer performance degradation due to the addition of extra modules. On the contrary, the model outperforms other module combinations on all three datasets, exhibiting superior stability and recognition capabilities. This indicates that the integration of the DualCascadeTSF module with MobileNetV2 not only enhances the model's performance but also effectively avoids the negative impacts associated with increased structural complexity.

### 4.3.3. Comparative Experiments with Classic Models

In order to verify the advantages and disadvantages of the model proposed in this paper, we selected several classic behavior recognition models for comparative experiments. All the models were trained and tested on the three datasets mentioned above (Crowd Violence, RWF-2000, and Hockey Fights) under the same experimental environment. The comparative analysis was mainly carried out from four aspects: accuracy, the number of parameters, the occupied memory size, and the calculation speed. The results are shown in Table 2.

**Table 2.** Comparison of accuracy rates of DualCascadeTSF-MobileNetV2 with other classical models on the three datasets.

Model	Crowd Violence(%)	RWF-2000(%)	Hockey Fights(%)
ResNet-50 [19]	93.878	84	95.5
3D-CNN [20]	94.3	82.75	94.4
LRCN [21]	94.57	77	97.1
I3D [22]	88.89	85.75	97.5
MiNet-3D [23]	91.41	81.98	94.71
AR-Net [24]	95.918	87.3	97.2
TSM [1]	95.95	88	97.5
TEA [25]	96.939	88.5	97.7
Two-cascade TSM [14]	96.939	89	98.05
SAM [26]	98.15	89.1	<b>99.1</b>
SSHA [27]	-	90.4	98.7
P-TSM [28]	96.969	<b>91</b>	98.5
MobileNet-TSM [18]	97.959	87.75	97.5
DualCascadeTSM-MobileNetV2(ours)	<b>98.98</b>	88.5	98.0

For the three indicators of the number of parameters, the occupied memory size, and the calculation speed, we used the third-party library torchsummary of PyTorch for calculation, and uniformly set the input parameters as a batch size of 8, the number of channels of 24, the height of the input video frame of 128, and the width of 128. The results are shown in Table 3.

**Table 3.** Comparison of the Number of Parameters, Memory Size, and Training Speed between DualCascadeTSF-MobileNetV2 and Other Classic Models.

Model	Parameters(MB)	Training Speed(min.)	Memory Size(MB)
3D-CNN [20]	297.83	248.38	2647.7
LRCN [21]	237.83	185.12	1212.93
I3D [22]	46.88	123.8	1000.2
TSN [29]	89.69	46.68	390.05
TSM [1]	89.69	113.43	397.71
TEA [25]	91.95	190.8	479.78
Two-cascade TSM [14]	89.69	128.33	397.71
P-TSM [28]	89.69	87.63	396.57
MobileNet-TSM [18]	8.49	32.68	175.86
DualCascadeTSM-MobileNetV2(ours)	16.99	35.92	347.13

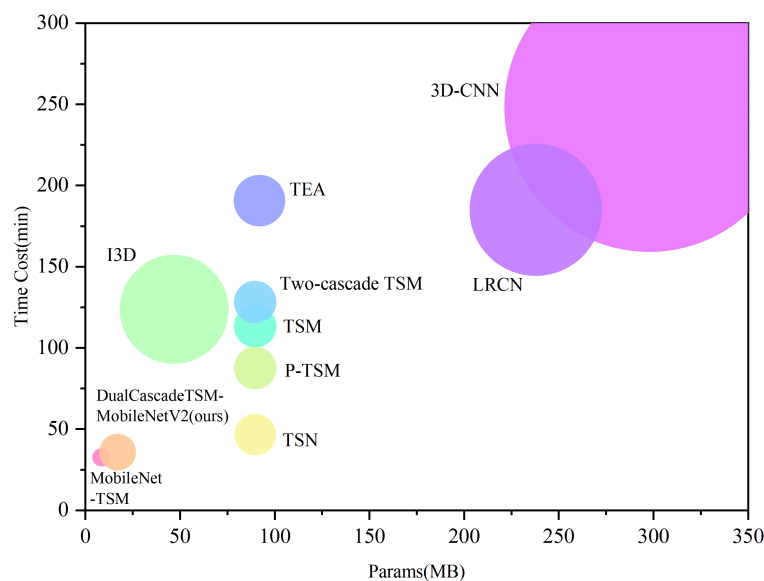
## 5. Discussion

As can be seen in Table 2, the DualCascadeTSF-MobileNetV2 model proposed in this paper has achieved results that are close to or even better than those of other classic models in the three violent behavior datasets. On the Crowd Violence dataset, the accuracy of DualCascadeTSF-MobileNetV2 reaches 98.98%, which is higher than other classical models and represents an improvement of 0.83% compared to the previously best-performing model, SAM. On the RWF-2000 dataset, its accuracy was 88.5%, higher than 8 models and slightly lower than 4 models, with only a 2.5% gap compared to the



highest-accuracy model, P-TSM. On the Hockey Fights dataset, its accuracy was 98.0%, higher than 9 models and slightly lower than 4 models, with only a 1.1% gap compared to the highest-accuracy model, SAM. Compared to MobileNet-TSM, which also uses MobileNetV2 as the base model, the accuracy of the model in this paper is 1.021%, 0.75%, and 0.5% higher on the three datasets, respectively. Compared with the lightweight model MiNet-3D, the accuracy of the model in this paper is 7.57%, 6.52%, and 3.29% higher on the three datasets respectively. Overall, although there is still a certain gap between DualCascadeTSF-MobileNetV2 and the most advanced algorithms in terms of accuracy, its performance is better than that of most models, especially among lightweight models, and it is significantly better than MobileNet-TSM and MiNet-3D.

To more intuitively demonstrate the performance of the models, this paper presents a comparison of DualCascadeTSF-MobileNetV2 with other classical models in terms of number of parameters, memory size, and training speed in Figure 11. In the figure, the x-axis represents the number of parameters, the y-axis represents the training time, and the area of the circle indicates the memory size of the model. The colors are used solely to distinguish different models and do not carry any other special meanings. As shown in the figure, the proposed model has a parameter quantity of 16.99 MB, a training time of 35.92 minutes, and a memory usage of 347.13 MB. Except for being slightly larger in parameter quantity and memory usage compared to MobileNet-TSM, the proposed model outperforms the other classical models in these three metrics, successfully achieving the design goal of being "lighter and faster" and demonstrating feasibility for deployment on edge devices.



**Figure 11.** Comparison of DualCascadeTSF-MobileNetV2 with other classical models in terms of the number of parameters, memory size, and training speed.

## 6. Conclusion

To meet the current demand for lightweight and efficient networks in violent behavior recognition tasks, based on the previous research results, this paper designs the DualCascadeTSF-MobileNetV2. The experimental results show that this model demonstrates excellent performance on the three datasets, namely Crowd Violence, Hockey Fights, and RWF-2000. Especially on the Crowd Violence dataset, its accuracy exceeds that of other classic models. In the other two datasets, although the accuracy does not reach the highest level, considering that this model has fewer parameters, a smaller size, and a faster operation speed, these advantages make its deployment on edge devices highly attractive. In practical applications, these performance optimizations can almost make up for the small accuracy gap. Therefore, the DualCascadeTSF-MobileNetV2 model has broad application prospects in future intelligent recognition edge devices.

Finally, this paper proposes a question worthy of further exploration: Literature[14] has pointed out that cascading more TSM modules within each residual block does not further enhance the model's feature extraction capabilities. The likely reason is that excessive temporal shifting operations make the connections between originally adjacent feature maps sparse. In this paper, we have enhanced the temporal coupling between feature maps by introducing feature fusion of the feature maps before and after each shift. So, if we further cascade more temporal shifting and feature fusion modules within each linear bottleneck layer of this model, will it further improve the model's feature extraction capabilities? We welcome readers to discuss this question with us!

**Supplementary Materials:** The following supporting information can be downloaded at the website of this paper posted on Preprints.org.

**Author Contributions:** Conceptualization, Y.L.(Yong Li) and Y.C.(Yuang Chen); methodology, Y.C.(Yuang Chen); software, Y.C.(Yuang Chen) and S.L.(Shaohua Li); validation, Y.C.(Yuang Chen), F.L.(Fang Li) and S.L.(Shuhan Lv); formal analysis, Y.C.(Yuang Chen); investigation, F.L.(Fang Li); resources, Y.L.(Yong Li); data curation, S.L.(Shuhan Lv); writing—original draft preparation, Y.C.(Yuang Chen); writing—review and editing, Y.L.(Yong Li); visualization, S.L.(Shaohua Li); supervision, Y.L.(Yong Li); project administration, Y.L.(Yong Li); funding acquisition, Y.L.(Yong Li). All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by Science and Technology Innovation Project grant number No. ZZKY20222304.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The data that support the findings of this study are openly available in the Crowd Violence dataset at <https://www.openup.ac.il/home/hassner/data/violentflows>, the RWF-2000 dataset at <https://github.com/mchengny/RWF2000-Video-Database-for-Violence-Detection>, and the Hockey Fights dataset at [https://www.reddit.com/r/datasets/comments/blath7/hockey\\_fight\\_dataset\\_availability](https://www.reddit.com/r/datasets/comments/blath7/hockey_fight_dataset_availability).

**Conflicts of Interest:** The authors declare no conflicts of interest.

## Abbreviations

The following abbreviations are used in this manuscript:

DualCascadeTSF	Dual Cascade Temporal Shift and Fusion module.
TSM	temporal shift module

## References

1. Lin, J.; Gan, C.; Han, S. TSM: Temporal Shift Module for Efficient Video Understanding. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*; Seoul, South Korea, 27 October–5 November 2019.
2. Liu, Z.; Li, J.; Shen, Z.; Huang, G.; Yan, S.; Zhang, C. Learning Efficient Convolutional Networks through Network Slimming. In *2017 IEEE International Conference on Computer Vision (ICCV)*; Venice, Italy, 22–29 October 2017.
3. Li, H.; Kadav, A.; Durdanovic, I.; Samet, H.; Graf, M. P. Pruning Filters for Efficient ConvNets. *CoRR* 2016, arXiv:1608.08710.
4. Sanh, V.; Wolf, T.; Rush, A. M. Movement Pruning: Adaptive Sparsity by Fine-Tuning. *arXiv* 2020, arXiv:2005.07683.
5. Howard, A. G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; others. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv preprint* 2017, arXiv:1704.04861.
6. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.-C. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*; Salt Lake City, UT, USA, 18–23 June 2018.
7. Zhang, X.; Zhou, X.; Lin, M.; Sun, J. ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*; Salt Lake City, UT, USA, June 18–23, 2018.

8. Iandola, F. N.; Han, S.; Moskewicz, M. W.; Ashraf, K.; Dally, W. J.; Keutzer, K. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size. *arXiv preprint arXiv:1602.07360* 2016.
9. Howard, A.; Sandler, M.; Chen, B.; Wang, W.; Chen, L.-C.; Tan, M.; Chu, G.; Vasudevan, V.; Zhu, Y.; Pang, R.; Adam, H.; Le, Q. Searching for MobileNetV3. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*; Seoul, Korea, Oct. 27 - Nov. 2, 2019.
10. Tan, M. and Le, Q. V., "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," in *Statistics*, vol. 2, 2019.
11. Wang, X.; Xiang, T.; Zhang, C.; Song, Y.; Liu, D.; Huang, H.; Cai, W. BiX-NAS: Searching Efficient Bi-directional Architecture for Medical Image Segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*; 2021.
12. Yan, B.; Peng, H.; Wu, K.; Wang, D.; Fu, J.; Lu, H. LightTrack: Finding Lightweight Neural Networks for Object Tracking via One-Shot Architecture Search. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*; 2021.
13. Cai, H.; Gan, C.; Han, S. Once for All: Train One Network and Specialize it for Efficient Deployment. *Statistics* **2019**.
14. Liang, Q.; Li, Y.; Chen, B.; Yang, K. Violence behavior recognition of two-cascade temporal shift module with attention mechanism. *Journal of Electronic Imaging* **2021**, 30(4), 43009.
15. Hassner, T.; Itcher, Y.; Kliper-Gross, O. Violent flows: Real-time detection of violent crowd behavior. In *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*; 2012.
16. Cheng, M.; Cai, K.; Li, M. RWF-2000: An Open Large Scale Video Database for Violence Detection. In *2020 25th International Conference on Pattern Recognition (ICPR)*; 2021.
17. Bermejo Nievas, E.; Deniz Suarez, O.; Bueno Garcia, G.; Sukthankar, R. Violence Detection in Video Using Computer Vision Techniques. In *International Conference on Computer Analysis of Images and Patterns*; 2011.
18. Zhang, Y.; Li, Y.; Guo, S. Lightweight mobile network for real-time violence recognition. *PloS one* **2022**, 17(10), e0276939.
19. Shafiq, M.; Gu, Z. Deep Residual Learning for Image Recognition: A Survey. *Applied Sciences* **2022**, 12, 8972.
20. Ji, S.; Xu, W.; Yang, M.; Yu, K. 3D convolutional neural networks for human action recognition. *IEEE Transactions On Pattern Analysis And Machine Intelligence* **2013**, 35(1), 221–231.
21. Donahue, J.; Hendricks, L. A.; Rohrbach, M.; Venugopalan, S.; Guadarrama, S.; Saenko, K.; Darrell, T. Long-Term Recurrent Convolutional Networks for Visual Recognition and Description. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **2017**, 39(4), 677–691.
22. Carreira, J.; Zisserman, A. Quo Vadis, action recognition? A new model and the kinetics dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*; 2017.
23. Wang, W.; Dong, S.; Zou, K.; Li, W. A Lightweight Network for Violence Detection. In *ICIGP 2022: 2022 the 5th International Conference on Image and Graphics Processing (ICIGP)*; 2022.
24. Meng, Y.; Lin, C.-C.; Panda, R.; Sattigeri, P.; Karlinsky, L.; Oliva, A.; Saenko, K.; Feris, R. AR-Net: Adaptive Frame Resolution for Efficient Action Recognition. In *Computer Vision – ECCV 2020*; 2020.
25. Li, Y.; Ji, B.; Shi, X.; Zhang, J.; Kang, B.; Wang, L. TEA: Temporal Excitation and Aggregation for Action Recognition. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*; 2020.
26. Asad, M.; Jiang, H.; Yang, J.; Tu, E.; Malik, A. A. Multi-Level Two-Stream Fusion-Based Spatio-Temporal Attention Model for Violence Detection and Localization. *International Journal of Pattern Recognition & Artificial Intelligence* **2022**, 36(1), 1–25.
27. Mohammadi, H.; Nazerfard, E. Video violence recognition and localization using a semi-supervised hard attention model. *Expert Systems with Applications* **2023**, 212, 118791.
28. Zhang, Y.; Li, Y.; Guo, S.; Liang, Q. Not all temporal shift modules are profitable. *Journal of Electronic Imaging* **2022**, 31(4), 043030.
29. 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. In *Computer Vision – ECCV 2016, Part VIII*; 2016; Vol. 9912, pp 20–36.

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.