Journal of Software Engineering and Applications, 2021, 14, 172-188
https://www.scirp.org/journal/jsea
ISSN Online: 1945-3124
ISSN Print: 1945-3116

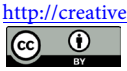
Action Recognition Using Multi-Scale Temporal Shift Module and Temporal Feature Difference Extraction Based on 2D CNN

Kun-Hsuan Wu, Ching-Te Chiu*

Department of Computer Science, National Tsing Hua University, Taiwan
Email: *grace.ct.chiu@gmail.com

How to cite this paper: Wu, K.-H. and Chiu, C.-T. (2021) Action Recognition Using Multi-Scale Temporal Shift Module and Temporal Feature Difference Extraction Based on 2D CNN. *Journal of Software Engineering and Applications*, **14**, 172-188.
https://doi.org/10.4236/jsea.2021.145011

Received: February 9, 2021
Accepted: May 24, 2021
Published: May 27, 2021



Abstract

Convolutional neural networks, which have achieved outstanding performance in image recognition, have been extensively applied to action recognition. The mainstream approaches to video understanding can be categorized into two-dimensional and three-dimensional convolutional neural networks. Although three-dimensional convolutional filters can learn the temporal correlation between different frames by extracting the features of multiple frames simultaneously, it results in an explosive number of parameters and calculation cost. Methods based on two-dimensional convolutional neural networks use fewer parameters; they often incorporate optical flow to compensate for their inability to learn temporal relationships. However, calculating the corresponding optical flow results in additional calculation cost; further, it necessitates the use of another model to learn the features of optical flow. We proposed an action recognition framework based on the two-dimensional convolutional neural network; therefore, it was necessary to resolve the lack of temporal relationships. To expand the temporal receptive field, we proposed a multi-scale temporal shift module, which was then combined with a temporal feature difference extraction module to extract the difference between the features of different frames. Finally, the model was compressed to make it more compact. We evaluated our method on two major action recognition benchmarks: the HMDB51 and UCF-101 datasets. Before compression, the proposed method achieved an accuracy of 72.83% on the HMDB51 dataset and 96.25% on the UCF-101 dataset. Following compression, the accuracy was still impressive, at 95.57% and 72.19% on each dataset. The final model was more compact than most related works.

Keywords

Action Recognition, Convolutional Neural Network, 2D CNN, Temporal Relationship

1. Introduction

Recently in the field of computer vision, human action recognition has become increasingly research-worthy. With the development of technology, action recognition has wide applications in the present era. Deep ConvNets, such as Inception-V1 [1], ResNet [2], and their variations [3] [4] [5] [6] have already achieved outstanding performance in image classification. Several studies on action recognition led to the direct inflation of the filters of these models from two-dimensional (2D) to three-dimensional (3D) to obtain inflated 3D ConvNets (I3D) [7], resolution 3D LLC (Res3D) [8], ResNeXt3D [9], among other models. Currently, there are two main approaches to action recognition: 2D CNN (convolutional neural network) and 3D CNN. The 2D CNN method performs convolution on one frame at a time, without temporal fusion. Conversely, the 3D CNN method performs convolution on multiple frames using 3D convolutional filters to achieve spatio-temporal learning.

In contrast to image recognition, video understanding requires learning the relevance of frames; therefore, the disadvantage of the 2D CNN method is its relatively limited performance when only RGB images are used for recognition. To improve their accuracy, most 2D CNN mainstream approaches, such as two-stream [10] and its variations [11] [12] [13], incorporate the optical flow field [14]; however, this leads to additional computational costs. Conversely, C3D [15], the mainstream method based on 3D CNN leverages the advantages of 3D convolutional kernels to perform convolution on multiple frames and effectively learn the correlation of adjacent sampled frames. However, compared with the 2D CNN, its architecture causes an explosion of parameters and calculations.

In this study, we aimed to further improve the performance of the traditional 2D CNN architecture for action recognition by expanding the temporal receptive fields. Although TSM [16] attempts to increase the temporal receptive fields to three, methods [11] [12] that generally use a stack of five optical flow frames as inputs with each RGB frame still have limitations. We propose the multi-scale temporal shift module (MSTSM), which can learn spatio-temporal information more effectively. Owing to the shift blocks with different scales, the entire model has higher receptive fields in the time dimension. Furthermore, many actions are prone to misprediction owing to the similarity of the movements that constitute them. The temporal feature difference extraction module of the proposed module can subtract the features of different frames to learn the uniqueness of the details of each action. **Figure 1** is a schematic diagram of the proposed model. Finally, we filtered out similar kernels to make the model more compact.

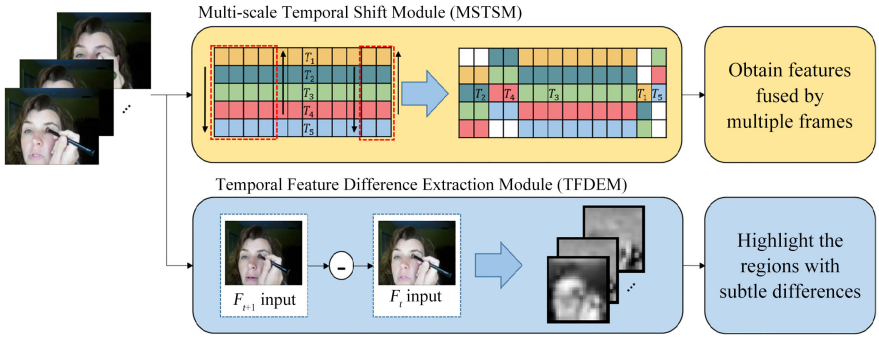


Figure 1. Schematic diagram of proposed method. Multi-scale temporal shift module merges features of multiple frames to enlarge the temporal receptive fields. Temporal feature difference extraction module subtracts features of different frames to highlight differences.

2. Related Works

Recently, many approaches to video understanding and action recognition have been proposed. We discuss some mainstream works in this section. Compared with the traditional methods [17] [18], these works can be broadly categorized into two classes: 2D ConvNets-based methods and 3D ConvNets-based methods.

2.1. 2D CNN

It is difficult to capture temporal relationship, which is crucial in video recognition, using methods based on 2D CNN; hence, most works incorporate other streams, such as optical flow or motion vector [19] [20] [21] [22], to compensate for this deficiency.

Simonyan *et al.* [10] designed a two-stream ConvNet framework that contains spatial and temporal streams. The input for the spatial stream is a still RGB image sampled from the source video, whereas that of the temporal stream is in the form of stacked dense optical flow. The outputs from these two streams are combined through late fusion to obtain the final prediction.

Wang *et al.* [11] proposed the TSN based on the aforementioned two-stream method. In this approach, the long-range temporal information is captured from the sampled frames using a sparse sampling strategy. First, the given input videos are sliced into several segments of equal length; then, one frame is sampled from each segment. Fusing the extracted features from these sampled snippets enable the framework to effectively learn the long-range relationships in the temporal dimension.

Lin *et al.* [16] proposed TSM, which propels the channel forward and backward along the temporal dimension; thus, the features of adjacent sampled frames are fused with the current frame after processing. It can be applied to any 2D ConvNet backbone to achieve a similar effect as 3D ConvNet without extra costs.

2.2. 3D CNN

Carreira *et al.* [7] proposed a framework called I3D, which inflates all the con-

volutional filters and pooling layers of the Inception-V1 model [1] from 2D kernels to 3D kernels. Because of the design of Inception-V1 and the pre-trained weights on the Kinetics dataset [23], this framework has fewer parameters compared to C3D [15] and thus circumvents the overfitting problem. However, the I3D samples frames from the whole video at the inference stage, which causes heavy computational cost.

Wang *et al.* [24] proposed an I3D-based framework that incorporated long short-term memory (LSTM) [25] for improved accuracy to model the high-level temporal features extracted by the Kinetics-pretrained I3D model. However, similar to the I3D, this approach experienced parameter explosion due to the LSTM.

T-C3D, a framework proposed by Liu *et al.* [26], first divides the given input video into three clips and samples eight frames from each clip. Thus, it can capture short-term features from frames in the same clips using 3D kernels and long-term features when fusing the prediction from each clip. Furthermore, Liu *et al.* employed compression methods [27] [28] to reduce the model size.

Although the compression technique reduces the model size, T-C3D still requires several frames for inference, which still causes explosive FLOPs.

3. Multi-Scale Temporal Shift Module and Temporal Feature Difference Extraction Based on 2D CNN for Action Recognition

We combined two proposed modules, MSTSM and temporal feature difference extraction module (TFDEM), on ResNet-50 [2], as shown in **Figure 2**. First, we adopted the sparse sampling strategy [11] and fed the sampled frames to our model. In the MSTSM, after shifting the feature maps along the temporal dimension, we replaced and concatenated different frame features in the two temporal shift blocks and increased the temporal receptive fields. In the TFDEM, to highlight the difference between the frames, we subtracted the feature maps of the current frame and the next frame. Then, we integrated the cross-entropy loss [29] value of the two paths to update the weights of the entire model. Finally, to make the model more compact, we filtered the kernels in the layers that had passed the MSTSM.

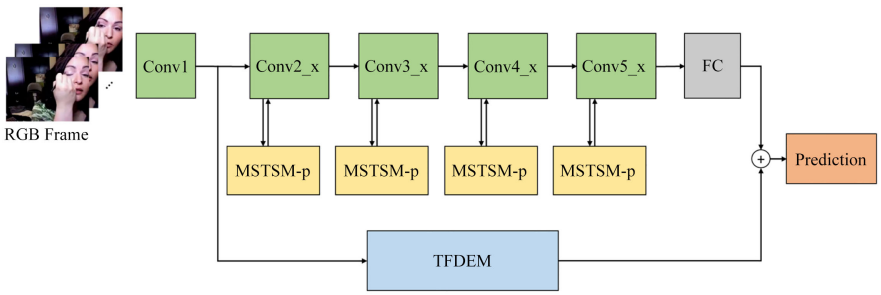


Figure 2. Overall architecture of proposed method. Backbone used was 2D ResNet-50; we performed convolution on different frames at different times. We pruned kernels of layers that passed MSTSM and denoted them as "MSTSM-p".

3.1. Sampling Strategy

Similar to Wang *et al.* [11], we adopted a sparse sampling strategy. The input video was sliced into several segments and one frame from each segment was sampled. This made it possible to understand the information conveyed by the entire video.

For example, given an input video with N frames, we divided the N frames into n parts of equal length; thus, each part was composed of $k = N/n$ frames. Then, the sampled frames from each part formed a set and were denoted as follows:

$$Frames = \{F_1, F_2, F_3, \cdots, F_n\} \tag{1}$$

where the frame number F_i for the training stage was a random number in the interval $[1,k]$ and plus $(i-1)*k$. For the testing stage, the frame number F_i was the median of the interval $[1,k]$ and plus $(i-1)*k$. In our experiments, n is 8, unless otherwise specified.

3.2. Multi-Scale Temporal Shift Module

As the name implies, the MSTSM shifts features with different scales along the temporal dimension. The main purpose of shifting the feature maps bi-directionally is to merge the features of different frames when performing convolution. **Figure 3** shows the detailed structure of the MSTSM. The rows with different colors indicate the features from different time units T_i, where i is the frame index. For brevity, we excluded the batch size, height, and width dimensions of the feature map. Next, we describe the benefits of the two temporal shift blocks with different shift units.

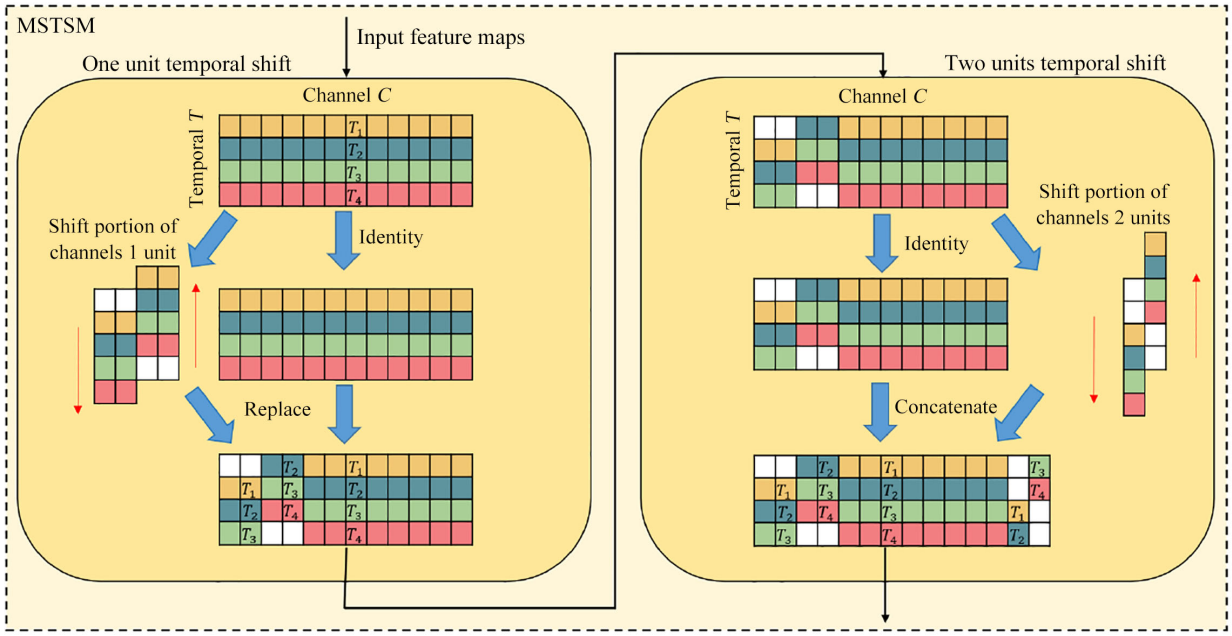


Figure 3. Detailed structure of proposed MSTSM. For input feature maps, rows with different colors indicate features from different time units T_i, where i is frame index.

1) *One-unit Temporal Shift Block*: In the case of shifting one unit, features are shifted from the frames before and after the current frame; this is also the relationship that needs to be learned most in action recognition. We selected the number of input channels with a higher ratio to be shifted. The intuitive idea was to replace the original feature maps with the shifted ones, as shown on the left side of Figure 3. However, the replaced feature map may contain very important channels from the original frame; thus, it may not be possible to learn the current frame effectively. Spatio-temporal learning can be achieved without incurring additional costs.

2) *Two-unit Temporal Shift Block*: After one-unit temporal shifting, to further increase the model's temporal receptive fields, we shifted the features by two units. Thus, we shifted the features of the frame before the previous frame and after the next frame to the current frame. Hence, as illustrated in Figure 4, we merged the information from the other four frames with those of the current frames, except for boundary cases.

To circumvent the aforementioned risk, we concatenated the shifted features with the original identical feature maps, as shown on the right side of Figure 3. Thus, the features with two-unit shifts could be considered extra information. However, although the information is critical, shifting a large number of channels with two units may confuse the model about the features that are close to the current frame; this may interfere with the learning of temporal order. Therefore, we selected the number of channels with a lower ratio in this case; this can also reduce computational costs.

3.3. Temporal Feature Difference Extraction Module

In several cases, the difference between the frames was subtle even though we adopted the sparse sampling strategy. Another problem was that the movements that constituted some actions differed only slightly. Thus, the proposed TFDEM module was designed to address these problems.

Figure 5 shows the detailed structure of the TFDEM. To maintain the efficiency and enlarge the receptive fields during feature extraction, the stride of the convolutional layers before subtraction was set to 2; this shrinks the size of the feature maps. However, subtracting the low-resolution features did not have significant impact; therefore, the bilinear upsample layer was inserted before feature subtraction. Then, the convolutional, global average pooling [30], and fully connected layers extracted and aggregated the subtracted features.

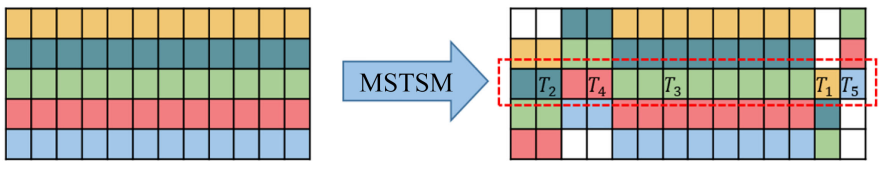


Figure 4. After passing through the proposed MSTSM, temporal receptive fields, except boundary cases, can be increased by five.

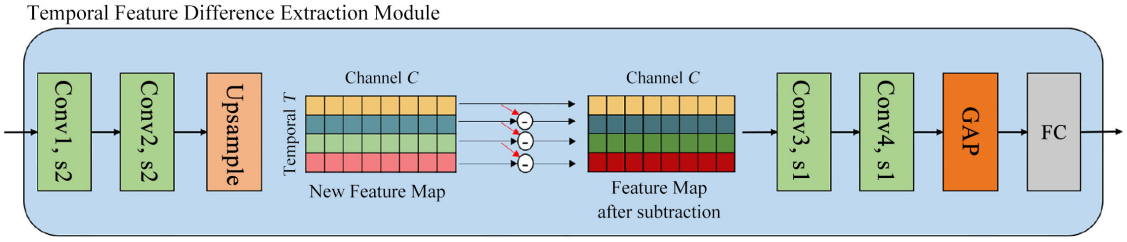


Figure 5. Schematic of proposed TFDEM (Temporal feature difference extraction module) that subtracts features of different frames to highlight differences.

Figure 6 shows the effect of our proposed TFDEM; the top row is the sampled frames from the given video. It is difficult to see the difference between the two sampled frames and the extracted features. The feature maps at the bottom row are the results following subtraction. Frame 1 (the bottom right) remains the same. The bottom left shows the result of subtracting the feature map of Frame 2 from that of Frame 1. It can be observed that the different regions between the frames are highlighted following subtraction.

3.4. Objective Function

We had two overall objective functions during the training stage: $\mathcal{L}_{main}$ and $\mathcal{L}_{TFDEM}$. The first objective function $\mathcal{L}_{main}$ was calculated based on the definition of the cross entropy with the output probability p_{main} from the main path and the ground-truth label y_{true}. The equation can be written as follows:

$$\arg\min \mathcal{L}_{main}\left(W_{main}\right) = \arg\min\left(-\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C}\left(y_{true}\right)^{i,c}\ln\left(p_{main}\right)^{i,c}\right) \tag{2}$$

where N is the total number of training videos, W_{main} is the learnable weight of the main path, y_{true} is the ground-truth label, C is the total number of categories, and $\left(p_{main}\right)^{i,c}$, produced by the main path, is the probability of the i^{th} video belonging to the c^{th} category.

Similarly, the equation of the second objective function $\mathcal{L}_{TFDEM}$ can be written as follows:

$$\arg\min \mathcal{L}_{TFDEM}\left(W_{TFDEM}\right) = \arg\min\left(-\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C}\left(y_{true}\right)^{i,c}\ln\left(p_{TFDEM}\right)^{i,c}\right) \tag{3}$$

where W_{TFDEM} is the learnable weight of the TFDEM path, and $\left(p_{TFDEM}\right)^{i,c}$, produced by the TFDEM path, is the probability of the i^{th} video belonging to the c^{th} category.

$\mathcal{L}_{total}$ is the sum of both objective functions. Hence, the equation can be written as follows:

$$\arg\min \mathcal{L}_{total}\left(W\right) = \arg\min \mathcal{L}_{main}\left(W_{main}\right) + \mathcal{L}_{TFDEM}\left(W_{TFDEM}\right) \tag{4}$$

where W is the learnable weight of the entire model.

3.5. Pruning Method

The kernel space of each convolutional layer contains some "similar" kernels.

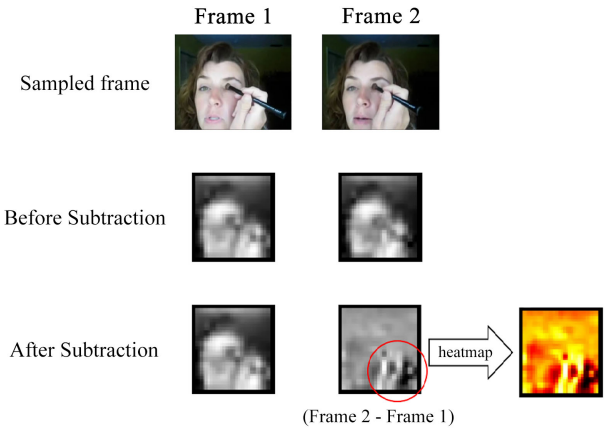


Figure 6. Example to visualize feature map before and after subtraction. **Top row:** sampled frames. **Middle row:** feature maps of each frame before subtraction. **Bottom row:** feature map of Frame 1 remains the same; other one is the difference of feature maps of Frames 1 and 2.

Performing convolution using similar kernels will result in similar outputs. Highly similar outputs may be redundant in the model; hence, some of them can be removed with no significant effect.

1) *Defining Similar Kernels*: We used two steps to define "similar" kernels. First, we searched for a kernel K^{GM} with the smallest distance from all other kernels in the kernel space; the equation can be written as follows:

$$td_i = \sum_{j \in [1, Channel_{out}]} \left\| K_i - K_j \right\|_2 \tag{5}$$

td_i is summation of the distance for a kernel K_i to all other kernels in the same kernel space.

$$K^{GM} = \min K_i, \tag{6}$$

where K_i with minimum td_i and $Channel_{out}$ is the number of output channels for each target layer and K_i is the i^{th} kernel in the kernel space. Second, we ranked all the kernels in the ascending order of their distance from K^{GM} using the following equation:

$$sort\left(d_1, d_2, \cdots, d_i, \cdots, d_{channel}\right),$$
$$d_i = \left\| K_i - K^{GM} \right\|_2, \forall\, i \in \left[1, Channel_{out}\right] \tag{7}$$

Then, we selected kernels according to the pruning ratio.

Therefore, the "similar kernels" referred to hereafter are kernels with a smaller distance from K^{GM}.

2) *Target Layers*: The target layers in our method were the layers with the MSTSM because the kernel similarity may either result in redundant spatial or temporal features, following shifting. Pruning this layer can filter out redundant spatial and temporal features, as illustrated in **Figure 7**.

3) *Averaging Selected Input Feature Maps*: It is known that each input feature map of the layer L_i is the output feature map of L_{i-1}. Therefore, we first found $C_{out}^{i-1} * 1/R_{in}$ similar kernels from the kernel space of L_{i-1}. Then, we averaged

the corresponding channels of the output feature map into one channel, where R_{in} is the prune ratio in this case and C_{out}^{i-1} is the number of output channels of L_{i-1}.

4) *Pruning Selected Kernels*: To prune the kernels in layer L_i, we found and eliminated similar kernels with the ratio R_{out}; therefore, both C_{out}^{i} and C_{in}^{i+1} reduced $C_{out}^{i} * 1/R_{out}$ channels, respectively, where C_{out}^{i} was the number of output channels of L_i and C_{in}^{i+1} was the number of input channels of L_{i+1}.

Both cases are illustrated in Figure 8.

4. Results

In this section, we first describe our experimental environment and datasets. Then, we present the experimental results to demonstrate that the proposed modules can improve the performance. Further, the ablation studies are described to show how we determined the optimal settings. At the end of this section, we present the comparison results of the proposed method and several state-of-the-art methods.

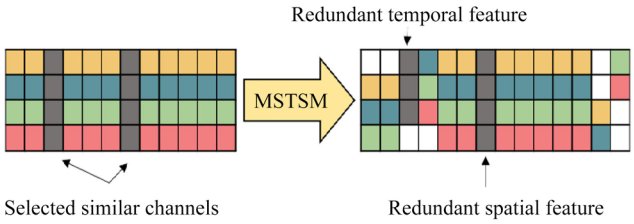


Figure 7. Target layer redundant features: After shifting, channels with high similarities form redundant spatial or temporal features.

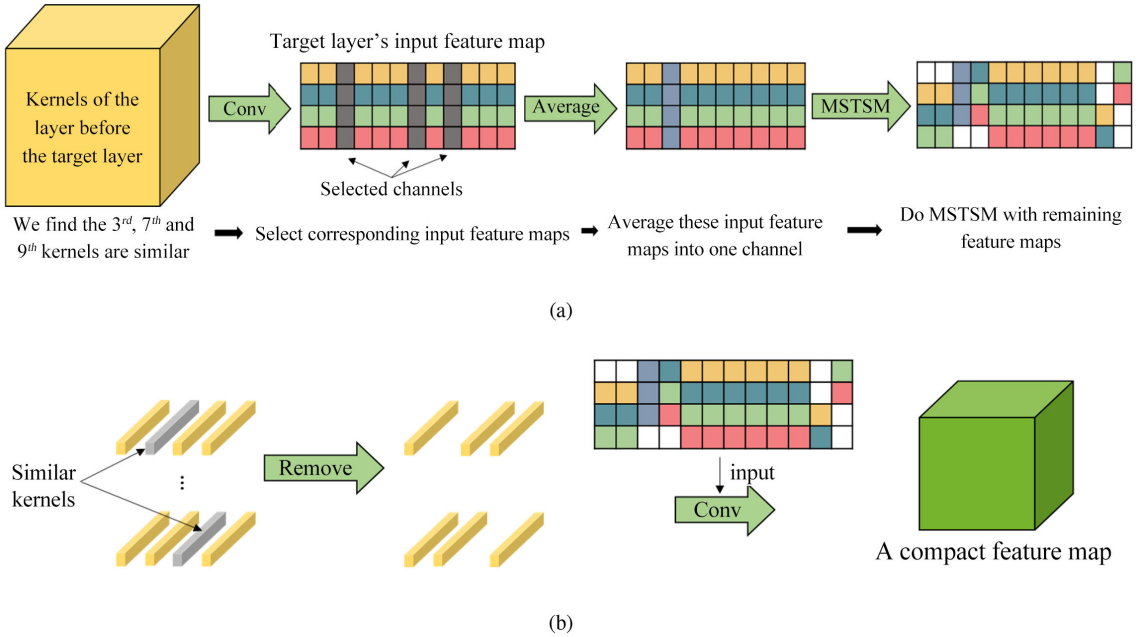


Figure 8. Selecting pruning channels. Example of (a) average input feature map based on the similar kernels of the previous layer. (b) Similar kernels in target layer were eliminated and convolution was conducted with averaged input feature maps.

4.1. Experimental Environment

We implemented our proposed method using the Pytorch [32] framework. Owing to the benefits of transfer learning, we pre-trained our model on the ImageNet [33] and Kinetics [23] datasets. During the fine-tuning process, we froze the batch normalization [3] layer. We trained our model with a weight decay of 0.0005, and an initial learning rate of 0.00025, which was divided by ten every ten epochs; the batch size was 16, and the optimizer was the stochastic gradient descent (SGD) [34]. The CPU was an Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.1 GHz, and the GPU was NVIDIA TITAN V.

4.2. Datasets

We mainly evaluated our framework on the UCF-101 dataset provided by [35] and the HMDB51 dataset provided by [36]. We briefly introduce these two datasets in the following sections.

1) *UCF*-101 [35]: The UCF-101 is one of the most popular datasets for action recognition. It contains 13,320 realistic action videos from 101 categories in five groups; most of the videos were collected from YouTube. The video frame rate in this dataset is 25 frames per second and the resolution 320 × 240. The dataset was composed of three training and testing splits; we also presented the average accuracy of the three splits, similar to the majority of existing works.

2) *HMDB*51 [36]: HMDB51 is another popular dataset used for evaluation in most action recognition research. The videos in this dataset were gathered mostly from movies. It consists of 6766 videos with 51 categories, each of which contains a minimum of 101 clips. The resolution of the videos in this dataset is 340 × 256; the frame rate was 30 frames per second. The dataset was composed of three training and testing splits. Similar to the evaluation strategy of the UCF-101 dataset, we presented the average accuracy of the three splits.

4.3. Data Augmentation

To improve the performance of the model and avoiding overfitting, we applied similar data augmentation as [16] during training. First, we resized the raw image to make the shorter side equal to 256 pixels. Then, we performed corner and center cropping on the height and width to satisfy the size $\in [256, 224, 192, 168]$. Subsequently, we resized the cropped image to 224 × 224. In the last step, a random horizontal flipping with probability $p = 0.5$ was performed.

4.4. Experimental Results and Ablation Studies

In this section, we present the results of the proposed methods step by step. Unless otherwise specified, the results were evaluated on the UCF-101 dataset.

We show the various settings of the proposed modules and discuss their influence on different aspects. Furthermore, we also attempted to apply our proposed method to a different backbone; we obtained competitive results. We also demonstrate the possibility of incorporating the optical flow into our method to

achieve greater accuracy.

Table 1 shows the comparison result of the baseline and proposed MSTSM. In the baseline, the uni-scale temporal shift module was combined with the backbone. Conversely, the proposed method fused the MSTSM with the backbone, which is denoted as MSTSM. From Table 1, it can be observed that the MSTSM improved the average accuracy by 0.71%.

Based on the results shown in Table 1, we incorporated the TFDEM into the MSTSM and denoted it as the MSTSM-TFDEM. Compared to the architecture with only the MSTSM, the average accuracy increased by 0.61%. Therefore, the results obtained using these two proposed modules were 1.32% more accurate, compared with the result yielded by the baseline.

Then, we applied our pruning method to the architecture based on the MSTSM and TFDEM and denoted it as MSTSM-TFDEM-p. After filtering out similar kernels, we preserved the useful and informative features. Based on the experimental result, we reduced approximately 2M of parameters and maintained an accuracy of 95.57%.

The influence of each module is summarized in Table 1. Regardless of whether we combined the TFDEM and MSTSM, the accuracy was significantly higher than that of the baseline. The highest accuracy was achieved by the combination of the modules.

1) *Incorporation with Optical Flow*: Similar to other works, we also attempted to incorporate the optical flow stream into the proposed module to achieve accuracy. The resultant architecture is shown in Table 1. The architecture with the optical flow stream is denoted as MSTSM-TFDEM-OF. We combined the RGB and optical flow networks at a weight ratio of 1:1.5, which is the best weight ratio based on our experiments. The accuracy of the module incorporating the optical flow was 1.6% higher.

Table 1. Experimental results of our proposed methods on UCF-101 dataset using ResNet-50 and efficientnet-B0 [31] as the backbone.

Backbone	Method	Accuracy	# Params	GFLOPs
ResNet-50	[16] (baseline) (our reimplementation)	94.93%	23.7	3.8
ResNet-50	MSTSM	95.64%	24.2	3.9
	[16] + TFDEM	95.71%	23.9	3.9
	MSTSM + TFDEM	96.25%	24.5	4.0
	MSTSM + TFDEM + pruning	95.57%	22.4	3.6
	MSTSM-TFDEM-OF	97.98%	49.0	7.9
EfficientNet-B0	MSTSM	95.60%	4.23	0.7
	MSTSM + TFDEM	96.08%	4.49	0.7

2) *Application to Different Backbones*: In this section, we demonstrate that our proposed modules can be applied to any backbone. We take the Efficient-Net-B0 [31] as an example. From the experimental results shown in **Table 1**, we obtained a similar accuracy as ResNet-50 and successfully compressed the number of parameters from 23.71 M to 4.49 M, which reduces the model size by approximately 81.06%.

3) *Shift Ratio of Each Temporal Shift Block*: Regarding the experiments, the first aspect to be discussed is the shift ratio of each temporal shift block. We experimented with various combinations of the ratio of each temporal shift block. The ratios of the one-unit temporal shift block and two-unit temporal shift block are denoted as R_{one} and R_{two}, respectively. As shown in **Table 2**, this can be divided into two cases: $R_{one} < R_{two}$ and $R_{one} > R_{two}$. In the first case ($R_{one} < R_{two}$), we set $R_{one} = 1/16$ and $R_{two} = 1/8$. The result was not satisfactory because, as mentioned in Section 3.2.1, the frame before and after the current frame is the most important relationship that must be learned to perform action recognition; hence, R_{one} should be greater than R_{two}. In another case ($R_{one} > R_{two}$), we fixed $R_{one} = 1/8$ and R_{two} varies from 1/12 to 1/20. With the decrease in R_{two}, the accuracy and number of parameters also decreased. To ensure higher accuracy and fewer parameters, we set $R_{one} = 1/8$ and $R_{two} = 1/16$.

4) *Temporal Receptive Fields of ConvNet*: In this section, we discuss the impact of the temporal receptive field (TRF). In addition to the two-unit temporal shift block, we also experimented with other scales, as shown in **Figure 9**. **Figure 9(a)** illustrates a variation of the two-unit-shift, which pads the features of the next frame to the vacancy bi-directionally. **Figure 9(b)** is the schematic of the MSTSM with three scales ranging from a one-unit-shift to a three-unit-shift.

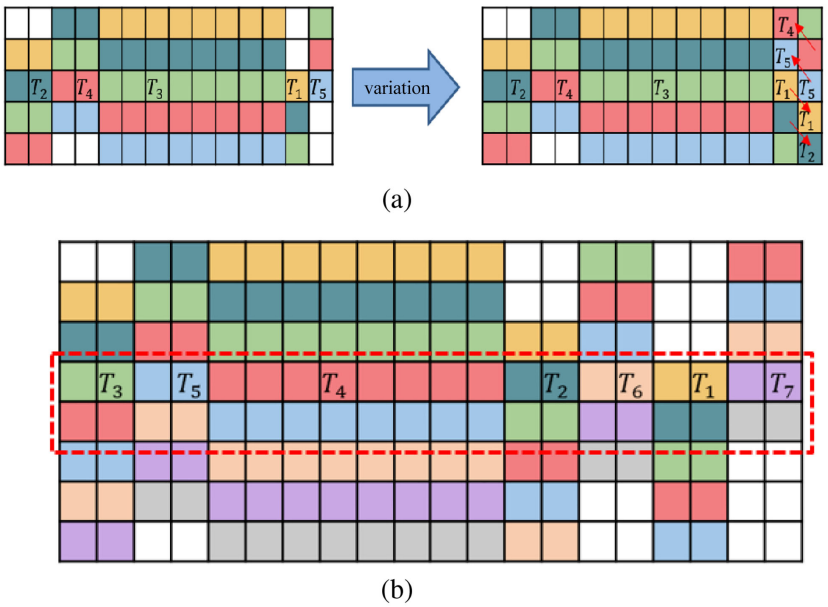


Figure 9. Different approaches to obtain higher TRF. (a) Variant of two-unit shift; moving next frame to pad vacancy. (b) Shift with three scales.

Table 2. Finding balance between accuracy and parameters tradeoff from ablation studies on various ratios of each temporal shift block.

	$R_{one} < R_{two}$				$R_{one} > R_{two}$			
	1/16	1/8	1/8	1/12	1/8	1/16	1/8	1/20
Accuracy	94.58%		95.79%		95.72%		95.26%	
# Params (M)	24.79		24.43		24.25		24.14	

In **Table 3**, $MSTSM_i$ indicates that we shifted the feature maps using i scale(s), and $MSTSM_{2var}$ denotes the variation shown in **Figure 9(a)**. From the experimental results shown in **Table 3**, the value of the temporal receptive field is not a fixed number because there are boundary cases. Furthermore, as the temporal receptive field increased, the corresponding accuracy did not necessarily increase. This is because some input videos have a large number of frames, causing the interval of our sampled frames to be so large that frames that were far away provided less relevant features. Therefore, we adopted the $MSTSM_2$ as our proposed module.

5) *Effect of Minimizing* L_{TFDEM}: Although subtracting the feature maps of different frames can highlight the difference between them, our TFDEM can still extract features inaccurately. Hence, we also minimized the loss value of the proposed TFDEM path to update the weights in it. We present the comparison result between the performance of the TFDEM with and without minimizing the loss value of the TFDEM path in **Table 4**.

4.5. Comparison with Other Works

We evaluated our framework on the UCF-101 and HMDB51 datasets and compared the performance with other modules in this section. As shown in **Table 5**, the proposed method outperformed the others while using only the RGB modality and eight frames.

Table 3. Comparison of architecture with different temporal receptive fields in terms of accuracy on UCF-101 dataset split-1.

Type of MSTSM	Accuracy	Temporal Receptive Fields
$MSTSM_1$	95.00%	2 - 3
$MSTSM_2$	95.72%	3 - 5
$MSTSM_{2var}$	95.20%	4 - 5
$MSTSM_3$	95.16%	4 - 7

Table 4. Comparison of accuracy of TFDEM with and without minimized loss value of TFDEM path on UCF-101 dataset.

Method	Accuracy
w/ L_{TFDEM}	96.25%
w/o L_{TFDEM}	95.59%

Table 5. Comparison of accuracy and number of parameters between proposed model and state-of-the-art methods on UCF101 dataset and HMDB51 dataset.

Works	Architecture	Modality	Sampling frames	# Params	Accuracy (UCF101)	Accuracy (HDMB51)
I3D-LSTM [24] (IOP'19)	3D CNN	RGB	whole video	-	95.1%	-
STH [20] (VCIP'19)	3D and 2D CNN	RGB, MV	16	88	94.3%	68.6%
T-C3D [26] (TCSVT'20)	3D CNN	RGB	24	31.7	92.5%	62.4%
IP-LSTM [37] (Access'20)	LSTM	RGB, OF	25	27.6	91.4%	68.2%
STDDCN [38] (PR'19)	2D CNN	RGB, OF	25	59	94.8%	69.49%
Heterogeneous Two-Stream [12]	2D CNN	RGB, OF	25	45.5	94.4%	67.2%
LVR [39] (ICMLA'19)	2D CNN	RGB, OF	25	92.8	94.4%	71.0%
Multi-teacher KD [21] (JSA'20)	2D CNN	RGB, MV, Residual	(1 + 11)	33.6	88.5%	56.16%
TSM [16] (ICCV'19)	2D CNN	RGB	8	23.7	94.9%	70.91%
TSN [11] (TPAMI'19)	2D CNN	RGB, OF	25	22.6	94.9%	71.0%
MSTSM-TFDEM (ours)	2D CNN	RGB	8	24.5	96.25%	72.83%
MSTSM-TFDEM-p (ours)	2D CNN	RGB	8	22.4	95.57%	72.19%
MSTSM-TFDEM (ours EfficientNet)	2D CNN	RGB	8	4.5	96.08%	72.48%

5. Conclusion

In this study, we designed an action recognition framework based on 2D ConvNet. When an RGB image alone is used as the 2D ConvNet input, there will be no information regarding the temporal relationship. To expand the temporal receptive fields without increasing the number of parameters, we proposed an MSTSM with average-sized receptive fields to learn the features from other frames. We also proposed a TFDEM to avoid mispredictions in the case of similar actions. Further, our pruning method made it possible to filter out similar kernels and obtain a compact model. Experimental results show that both the MSTSM and TFDEM are effective and our modules can effortlessly be applied to any other backbone. With both proposed modules, it was possible to achieve an accuracy of 96.25% on the UCF-101 dataset, which is a 1.1% improvement on the I3D-LSTM [24]. For the HMDB51 dataset, we achieved 72.83% accuracy, an improvement of 1.8% on the LVR [39]. After compression, the number of parameters was effectively reduced by approximately 2M, and an accuracy of 95.57% and 72.19% was achieved on the UCF-101 and HMDB51 datasets, respectively.

Conflicts of Interest

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

References

[1] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S.E., Anguelov, D., Erhan, D., Vanhoucke, V. and Rabinovich, A. (2015) Going Deeper with Convolutions. *IEEE*

Conference on Computer Vision and Pattern Recognition, *CVPR* 2015, Boston, 7-12 June 2015, 1-9. https://doi.org/10.1109/CVPR.2015.7298594

[2] He, K., Zhang, X., Ren, S. and Sun, J. (2016) Deep Residual Learning for Image Recognition. 2016 *IEEE Conference on Computer Vision and Pattern Recognition*, *CVPR* 2016, Las Vegas, 27-30 June 2016, 770-778. https://doi.org/10.1109/CVPR.2016.90

[3] Ioffe, S. and Szegedy, C. (2015) Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *Proceedings of the* 32*nd International Conference on Machine Learning*, *ICML* 2015, Lille, 6-11 July 2015, 448-456. http://proceedings.mlr.press/v37/ioffe15.html

[4] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. and Wojna, Z. (2016) Rethinking the Inception Architecture for Computer Vision. 2016 *IEEE Conference on Computer Vision and Pattern Recognition*, *CVPR* 2016, Las Vegas, 27-30 June 2016, 2818-2826. https://doi.org/10.1109/CVPR.2016.308

[5] Xie, S., Girshick, R.B., Dollár, P., Tu, Z. and He, K. (2017) Aggregated Residual Transformations for Deep Neural Networks. 2017 *IEEE Conference on Computer Vision and Pattern Recognition*, *CVPR* 2017, Honolulu, 21-26 July 2017, 5987-5995. https://doi.org/10.1109/CVPR.2017.634

[6] Hu, J., Shen, L. and Sun, G. (2018) Squeeze-and-Excitation Networks. 2018 *IEEE Conference on Computer Vision and Pattern Recognition*, *CVPR* 2018, Salt Lake City, 18-22 June 2018, 7132-7141. https://openaccess.thecvf.com/content_cvpr_2018/html/Hu_Squeeze-and-Excitation_Networks_CVPR_2018_paper.html

[7] Carreira, J. and Zisserman, A. (2017) Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. 2017 *IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), Honolulu, 21-26 July 2017, 4724-4733. http://arxiv.org/abs/1705.07750 https://doi.org/10.1109/CVPR.2017.502

[8] Tran. D., Ray. J., Shou. Z., Chang, S. and Paluri, M. (2017) Convnet Architecture Search for Spatiotemporal Feature Learning. CoRR, abs/1708.05038. http://arxiv.org/abs/1708.05038

[9] Hara, K., Kataoka, H. and Satoh, Y. (2018) Can Spatiotemporal 3D CNNs Retrace the History of 2D CNNs and ImageNet? 2018 *IEEE Conference on Computer Vision and Pattern Recognition*, *CVPR* 2018, Salt Lake City, 18-23 June 2018, 6546-6555. https://openaccess.thecvf.com/content_cvpr_2018/papers/Hara_Can_Spatiotemporal_3D_CVPR_2018_paper.pdf https://doi.org/10.1109/CVPR.2018.00685

[10] Simonyan, K. and Zisserman, A. (2014) Two-Stream Convolutional Networks for Action Recognition in Videos. *Advances in Neural Information Processing Systems* 27: *Annual Conference on Neural Information Processing Systems* 2014, Montreal, 8-13 December 2014, 568-576. https://proceedings.neurips.cc/paper/2014/hash/00ec53c4682d36f5c4359f4ae7bd7ba1-Abstract.html

[11] Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X. and Gool, L.V. (2019) Temporal Segment Networks for Action Recognition in Videos. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **41**, 2740-2755. https://doi.org/10.1109/TPAMI.2018.2868668

[12] Chen, E., Bai, X., Gao, L., Tinega, H.C. and Ding, Y. (2019) A Spatiotemporal Heterogeneous Two-Stream Network for Action Recognition. *IEEE Access*, **7**, 57267-57275. https://doi.org/10.1109/ACCESS.2019.2910604

[13] Wang, Y., Long, M., Wang, J. and Yu, P.S. (2019) Spatiotemporal Pyramid Network for Video Action Recognition. CoRR, abs/1903.01038. http://arxiv.org/abs/1903.01038

[14] Pérez, J.S., Meinhardt-Llopis, E. and Facciolo, G. (2013) TV-L1 Optical Flow Estimation. *Image Processing on Line*, **3**, 137-150. https://doi.org/10.5201/ipol.2013.26

[15] Tran, D., Bourdev, L.D., Fergus, R., Torresani, L. and Paluri, M. (2015) Learning Spatiotemporal Features with 3D Convolutional Networks. 2015 *IEEE International Conference on Computer Vision*, *ICCV* 2015, Santiago, 7-13 December 2015, 4489-4497. https://doi.org/10.1109/ICCV.2015.510

[16] Lin, J., Gan, C. and Han, S. (2019) TSM: Temporal Shift Module for Efficient Video Understanding. 2019 *IEEE/CVF International Conference on Computer Vision*, *ICCV* 2019, Seoul, 27 October-2 November 2019, 7082-7092. https://doi.org/10.1109/ICCV.2019.00718

[17] Dalal, N. and Triggs, B. (2005) Histograms of Oriented Gradients for Human Detection. 2005 *IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (*CVPR* 2005), San Diego, 20-26 June 2005, 886-893.

[18] Wang, H. and Schmid, C. (2013) Action Recognition with Improved Trajectories. *IEEE International Conference on Computer Vision*, *ICCV* 2013, Sydney, 1-8 December 2013, 3551-3558. https://doi.org/10.1109/ICCV.2013.441

[19] Wu, C., Zaheer, M., Hu, H., Manmatha, R., Smola, A.J. and Krähenbühl, P. (2018) Compressed Video Action Recognition. 2018 *IEEE Conference on Computer Vision and Pattern Recognition*, *CVPR* 2018, Salt Lake City, 18-23 June 2018, 6026-6035. https://openaccess.thecvf.com/content_cvpr_2018/html/Wu_Compressed_Video_Action_CVPR_2018_paper.html https://doi.org/10.1109/CVPR.2018.00631

[20] Li, S., Zhao, Z. and Su, F. (2019) A Spatio-Temporal Hybrid Network for Action Recognition. 2019 *IEEE Visual Communications and Image Processing*, *VCIP* 2019, Sydney, 1-4 December 2019, 1-4. https://doi.org/10.1109/VCIP47243.2019.8965878

[21] Wu, M.-C. and Chiu, C.-T. (2020) Multi-Teacher Knowledge Distillation for Compressed Video Action Recognition Based on Deep Learning. *Journal of Systems Architecture*, **103**, Article ID: 101695. https://doi.org/10.1016/j.sysarc.2019.101695

[22] Zhang, B., Wang, L., Wang, Z., Qiao, Y. and Wang, H. (2018) Real-Time Action Recognition with Deeply Transferred Motion Vector CNNs. *IEEE Transactions on Image Processing*, **27**, 2326-2339. https://doi.org/10.1109/TIP.2018.2791180

[23] Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., Suleyman, M. and Zisserman, A. (2017) The Kinetics Human Action Video Dataset. CoRR, abs/1705.06950. http://arxiv.org/abs/1705.06950

[24] Wang, X., Miao, Z., Zhang, R. and Hao, S. (2019) I3d-LSTM: A New Model for Human Action Recognition. *IOP Conference Series: Materials Science and Engineering*, **569**, Article ID: 032035. https://doi.org/10.1088/1757-899X/569/3/032035

[25] Hochreiter, S. and Schmidhuber, J. (1997) Long Short-Term Memory. *Neural Computation*, **9**, 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735

[26] Liu, K., Liu, W., Ma, H., Tan, M. and Gan, C. (2020) A Real-Time Action Representation with Temporal Encoding and Deep Compression. CoRR, abs/2006.09675. https://arxiv.org/abs/2006.09675

[27] Han, S., Mao, H. and Dally, W.J. (2016) Deep Compression: Compressing Deep Neural Network with Pruning, Trained Quantization and Huffman Coding. 4*th In-*

ternational Conference on Learning Representations, *ICLR* 2016, San Juan, 2-4 May 2016. http://arxiv.org/abs/1510.00149

[28] Han, S., Pool, J., Narang, S., Mao, H., Gong, E., Tang, S., Elsen, E., Vajda, P., Paluri, M., Tran, J., Catanzaro, B. and Dally, W.J. (2017) DSD: Dense Sparse-Dense Training for Deep Neural Networks. 5*th International Conference on Learning Representations*, *ICLR* 2017, Toulon, 24-26 April 2017. https://arxiv.org/abs/1607.04381

[29] Rubinstein, R.Y. and Kroese, D.P. (2004) The Cross-Entropy Method: A Unified Approach to Monte Carlo Simulation, Randomized Optimization and Machine Learning. Information Science & Statistics, Springer Verlag, New York.

[30] Lin, M., Chen, Q. and Yan, S. (2014) Network in Network. 2*nd International Conference on Learning Representations*, *ICLR* 2014, Banff, AB, 14-16 April 2014. http://arxiv.org/abs/1312.4400

[31] Tan, M. and Le, Q.V. (2019) EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *Proceedings of the* 36*th International Conference on Machine Learning*, *ICML* 2019, Long Beach, 9-15 June 2019, 6105-6114. http://proceedings.mlr.press/v97/tan19a.html

[32] Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L. and Lerer, A. (2017) Automatic Differentiation in Pytorch. https://openreview.net/forum?id=BJJsrmfCZ

[33] Deng, J., Dong, W., Socher, R., Li, L., Li, K. and Li, F. (2009) ImageNet: A Large-Scale Hierarchical Image Database. 2009 *IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (*CVPR* 2009), Miami, 20-25 June 2009, 248-255. https://doi.org/10.1109/CVPR.2009.5206848

[34] LeCun, Y., Boser, B.E., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W.E. and Jackel, L.D. (1989) Back Propagation Applied to Handwritten Zip Code Recognition. *Neural Computation*, **1**, 541-551. https://doi.org/10.1162/neco.1989.1.4.541

[35] Soomro, K., Zamir, A.R. and Shah, M. (2012) UCF101: A Dataset of 101 Human Actions Classes from Videos in the Wild. CoRR, abs/1212.0402. http://arxiv.org/abs/1212.0402

[36] Kuehne, H., Jhuang, H., Garrote, E., Poggio, T.A. and Serre, T. (2011) HMDB: A Large Video Database for Human Motion Recognition. *IEEE International Conference on Computer Vision*, *ICCV* 2011, Barcelona, 6-13 November 2011, 2556-2563. https://doi.org/10.1109/ICCV.2011.6126543

[37] Yu, S., Xie, L., Liu, L. and Xia, D. (2020) Learning Long-Term Temporal Features with Deep Neural Networks for Human Action Recognition. *IEEE Access*, **8**, 1840-1850. https://doi.org/10.1109/ACCESS.2019.2962284

[38] Hao, W. and Zhang, Z. (2019) Spatiotemporal Distilled Dense-Connectivity Network for Video Action Recognition. *Pattern Recognition*, **92**, 13-24. https://doi.org/10.1016/j.patcog.2019.03.005

[39] Maia, H.A., Souza, M.R., Santos, A., Pedrini, H., Tacon, H., de Souza Brito, A., Chaves, H.D.L., Vieira, M.B. and Villela, S.M. (2019) Learnable Visual Rhythms Based on the Stacking of Convolutional Neural Networks for Action Recognition. 18*th IEEE International Conference on Machine Learning and Applications* (*ICMLA*), Boca Raton, 16-19 December 2019, 1794-1799. https://doi.org/10.1109/ICMLA.2019.00290