Action Recognition with Bootstrapping based Long-range Temporal Context Attention

Ziming Liu
Beijing Institute of Technology
Beijing, China
liuziming.email@gmail.com

Guangyu Gao*
Beijing Institute of Technology
Beijing, China
guangyugao@bit.edu.cn

A. K. Qin
Swinburne University of Technology
Melbourne, Australia
kqin@swin.edu.au

Tong Wu
Beijing Institute of Technology
Beijing, China
3220190896@bit.edu.cn

Chi Harold Liu
Beijing Institute of Technology
Beijing, China
chiliu@bit.edu.cn

ABSTRACT

Actions always refer to complex vision variations in a long-range redundant video sequence. Instead of focusing on limited range sequence, i.e. convolution on adjacent frames, in this paper, we proposed an action recognition approach with bootstrapping based long-range temporal context attention. Specifically, due to vision variations of the local region across frames, we target at capturing temporal context by proposing the Temporal Pixels based Parallel-head Attention (TPPA) block. In TPPA, we apply the self-attention mechanism between local regions at the same position across temporal frames to capture the interaction impacts. Meanwhile, to deal with video redundancy and capture long-range context, the TPPA is extended to the Random Frames based Bootstrapping Attention (RFBA) framework. While the bootstrapping sampling frames have the same distribution of the whole video sequence, the RFBA not only captures longer temporal context with only a few sampling frames but also has comprehensive representation through multiple sampling. Furthermore, we also try to apply this temporal context attention to image-based action recognition, by transforming the image into "pseudo video" with the spatial shift. Finally, we conduct extensive experiments and empirical evaluations on two most popular datasets: *UCF101* for videos and *Stanford40* for images. In particular, our approach achieves top-1 accuracy of 91.7% in UCF101 and mAP of 90.9% in Stanford40.

CCS CONCEPTS

• **Computing methodologies** → *Computer vision tasks.*

*Guangyu Gao is the Corresponding Author.
This work was supported by the National Natural Science Foundation of China under Grant No. U1736117, and in part by the Australian Research Council with Grant No. LP170100416 and LP180100114.

MM '19, October 21–25, 2019, Nice, France

ACM ISBN 978-1-4503-6889-6/19/10…$15.00
https://doi.org/10.1145/3343031.3350916

KEYWORDS

Action recognition, Context, self-attention, Bootstrapping attention

ACM Reference Format:
Ziming Liu, Guangyu Gao, A. K. Qin, Tong Wu, and Chi Harold Liu. 2019. Action Recognition with Bootstrapping based Long-range Temporal Context Attention . In *Proceedings of the 27th ACM International Conference on Multimedia (MM '19), October 21–25, 2019, Nice, France.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3343031.3350916

1 INTRODUCTION

Action recognition is a challenging task affected by many factors, such as different illumination conditions, diversity of perspectives, complex background, and large intra-class changes. Traditionally, the best performing algorithm was iDT (improved Dense Trajectories) [29], and the subsequent work was basically to improve the iDT method. With deep learning, there are more ways to solve the problem, including Two-Stream [31], C3D (Convolutional 3D) [6], and RNN [7]. Up to now, with complex vision variations, action recognition has not been thoroughly solved.

Modeling context dependencies is of crucial importance, especially for deep neural network-based vision tasks. In images or videos, traditional convolution operations with receptive fields are widely used to model the context in spatial space. Meanwhile, the recurrent operation is another widely used methods for the temporal context in sequential data. Most existed deep learning methods are essentially trying to involve the context dependency for better action recognition performance. However, most of them model the context dependency within a limited range. For long-range context, Wang et al. [32] proposed the non-local operation and block to model more general dependencies, and proved the effectiveness in most of the computer vision tasks, especially video-based tasks.

Furthermore, a video has a lot of redundancy, while the adjacent frames are always similar. For most action recognition methods, i.e. C3D[26] or LSTM[5], while they deal with a short range of similar frames, it is equivalent to operation on the single identity frame. Besides, if the video is extremely redundant, these methods not only improve limited accuracy but also the repeating operation will result in larger computation cost.

Meanwhile, an action is defined as some object interaction in some given scenario, thus, for more satisfactory recognition, we should pay attention to local features. [13] also suggested that the strong relations between objects contribute more to object detection.

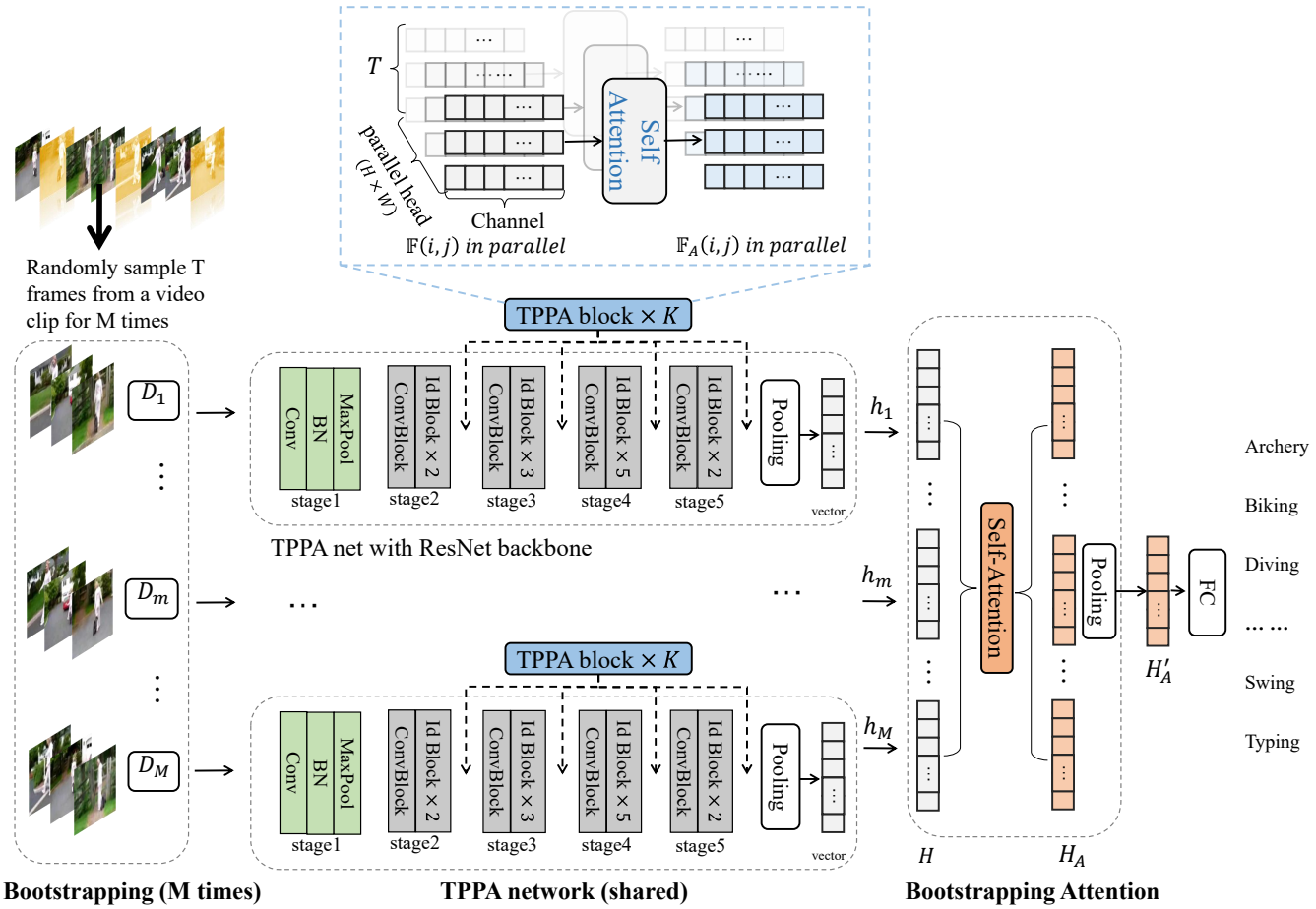

Figure 1: The overall network structure of the proposed RFBA framework

Zhao et al. [38] proposed a semantic part based action recognition method. Work[33] represents videos as space-time region graph based on object local features.

To capture the context-dependency more reasonably, we pay more attention to long-range context-dependency with self-attention. self-attention [28] computes the response at a position in a sequence by attending to all positions and taking their weighted average in an embedding space. Most of the related studies take the whole frame as a sequential position with very high computation complexity. Moreover, action recognition cares more on local objects, thus we defined the temporal pixels (all the pixels in the same spatial coordinate position across frames) as the basic sample for self-attention. Then, we designed the self-attention based *Temporal Pixels based Parallel-head Attention* (TPPA) block with the pre-defined temporal pixels. Besides, in TPPA, all the temporal pixels in different coordinate position are set as parallel multi-head to implement in parallel across the spatial dimension. The self-attention operation on temporal pixels has lower complexity compared to that on the whole of frames. The TPPA is a flexible block that can be added to any popular Deep Neural Networks.

For most action categories, the key features locate in a few key-frames, namely, the action can be recognized with a few frames. In order to capture these key frames in a longer-range context in a redundant video, we also designed the *Random Frames based Bootstrapping Attention* (RFBA) framework with TPPA. With the RFBA framework, we used the bootstrapping way[9] to re-sample the initial video many times. Then, these sampled frames are feed into the TPPA block and followed by another self-attention module for more comprehensive feature representation. The general flowchart of our paper can be seen in Fig. 1. Overall, the main contribution can be summarized as the following:

- The TPPA block is designed to capture the temporal context-dependency with self-attention. It not only archive better recognition but also has lower computation cost.
- To deal with video redundancy and capture more comprehensive representation, the RFBA framework is established on TPPA with bootstrapping-style key frame sampling.
- The proposed approach provided significant performance improvement from the state-of-the-art methods, and the visualization result also shows the rationality of it.

2 RELATED WORKS

Attention mechanism. Model with attention can pay attention to particular regions to emphasize important objects or regions.

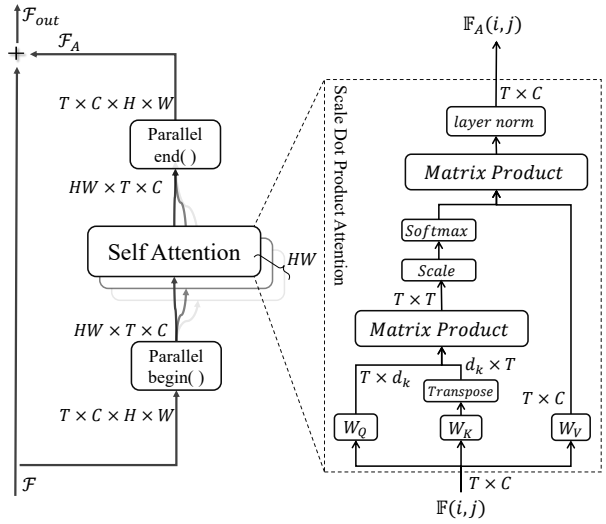

Figure 2: The structure of the TPPA block

Existing attention mechanisms can be grouped into hard attention, soft attention and self-attention. Mnih et al.[22] and Ba et al.[1] applied RNN to obtain hard attention map, which is hard binary.

Soft attention is obtained with continuous values, which means we can train the attention model easily. Sharama et al.[23] proposed a soft-attention LSTM method to pay more attention to some video frames, achieving better video classify performance. Li et al.[17] proposed Video LSTM to model soft attention. these methods take large computation cost, but only obtain a slight boost.

Another better attention mechanism with lower computational cost is self-attention. Vaswani et al.[28] summarized the previous self-attentive models into a unified form. The self-attention mechanism is first used in NLP and is introduced to the CV field. Hu et al.[13] tried to capture the relations of the object by apply self-attention mechanism in the object detection task. Wang et al.[32] proposed several different self-attention versions for image task. But all of these methods still apply attention to objects or frames, which are global features. Our proposed method is a new version of self-attention mechanism, which focuses on modeling attention in local features.

Action recognition. Before the era of deep learning, there have been some works to solve the human-centered computer vision tasks[21]. For action recognition, iDT (improved Dense Trajectories) is a typical work[29]. Recently, deep learning has achieved significant performance in many computer vision tasks[18, 19], there are also some works about action recognition with DNNs. Gkioxari et al.[11] proposed R* CNN based on Region-based Convolution Network(CNN)[10], which concentrate on model context of local features set. Zhao et al.[38] also achieved action classify by modeling local features of the human body. However, these methods fuse local features simply, without capturing the complex context of local features.

There are also many methods proposed to model spatiotemporal information. Simonyan et al.[24] proposed a new two-stream architecture by using the optical flow feature and RGB feature as input. However, optical flow only captures the local motion signals, TSN[30], TRN[40] is improved architectures based on two stream network[24]. C3D[26] is another typical architecture, which expends 2D CNNs into 3D CNNs. 3D convolution operation has the ability of modeling spatiotemporal information, but also limited to the kernel receptive field. Varol et al.[27] proved that 3D CNNs can achieve better results with the longer temporal length of input and using optical flow features. Although 3D CNNs achieve good performance, the computation cost is also larger.

Donahue et al.[5] introduced RNN to capture longer temporal information. Ng et al.[37] proposed two-stream LSTMs. However, the result of RNN based methods is not good enough, and RNN can't deal with longer video sequence well. Our method is designed to capture context information of the entire input sequence with lower computation cost and break the limitation of video length.

3 LONG-RANGE CONTEXT ATTENTION

3.1 Basic Attention Module

The long-range context attention is built on the self-attention in NLP [28], which is based on the Scaled Dot-Product Attention, as shown in Eq. 1. The input of the attention module includes the query $q \in \mathbf{Q}$ and all keys (stacked into the matrix $\mathbf{K}$) of dimension d_k, and values (stacked into the matrix $\mathbf{V}$) of dimension d_v. The similarity is obtained by the dot product between $\mathbf{Q}$ and $\mathbf{K}$. Given $\mathbf{Q}, \mathbf{K}, \mathbf{V}$, with the softmax function, an attention value is obtained by the weighted sum of the input values.

$$v_{out} = softmax(\frac{qK^T}{\sqrt{d_k}})V \tag{1}$$

Actually, self-attention is a special case of attention mechanism described above, where the **Q,K,V** are all the same as the input sequence $\mathbf{X} = \{x_i\}$. To derive the *video context* for action recognition by self-attention, we defined the **temporal pixels** at first. Temporal pixels means a set of $channel - dim$ vectors, $\mathbb{F}(i,j) = \{\mathcal{F}^t(i,j)\}_{t=1}^{T}$, and $\mathcal{F}$ is the input feature tensor shaped as $T \times C \times H \times W$, the set's size equals to the temporal size of input T, each vector $\mathcal{F}^t(i,j)$ consists of pixel values in the same spatial position (i,j) across channels. The input of the self-attention module in this paper is matrix $X = \mathbb{F}(i,j)$ as in Fig.2.

3.2 Temporal Pixels based Parallel-head Attention (TPPA)

There are some shortcoming to view the global frame(all spatial position) as a sample for self-attention module. For example, the dimension of the feature is extremely high, and it introduces a large computation cost. For example, $2048 \times 7^2 = 100,352$ is a typical feature dimension after ResNet-50 network. If we directly transform the feature to a lower dimension, i.e., with FC layer, it will lose too much information. Meanwhile, the action is always defined as several specific objects' interaction within some special scenario. That is to say, the local contents, i.e. objects or local regions, are very crucial for more accurate recognition. Besides, the above mentioned temporal pixels refer to a local region with the same semantic content. Therefore, in order to capture context dependency across different frames, we take the above mentioned

temporal pixels as the basic sample for self-attention processing, as shown in Fig.2.

The TPPA block is inserted after a CNN layer with output of C feature maps (size of $C \times H \times W$), i.e. a feature map set. Given the input $\mathcal{F}$ as T feature map sets from a video of T frames after a CNN layer. $\mathcal{F}^{(t)} \in \mathbb{R}^{C \times H \times W}$ refers to the t_{th} feature map set of $C \times H \times W$. A set of temporal pixels is notated as $\mathbb{F}(i,j) = \{\mathcal{F}^{(t)}(i,j,:)\}_{t=1}^{T}$, where $\mathcal{F}^{(t)}(i,j,:)$ (abbreviated as $\mathcal{F}^{(t)}(i,j)$) refers to a $C-dim$ vector with values in spatial position (i,j) across the t^{th} feature map set. When we defined $\mathcal{F}_A^{(t)}(i,j)$ as the output of self-attention on temporal pixels, the t^{th} *overall attention feature map set* will be $\mathbb{F}_A^{(t)} = \{\mathcal{F}_A^{(t)}(i,j)\}$. where $\mathcal{F}_A^{(t)}(i,j)$ can be computed as:

$$\mathcal{F}_A^{(t)}(i,j) = \sum_{n=1}^{T} \mathrm{w}^{nt}(i,j) \cdot (W_V \cdot \mathcal{F}^{(n)}(i,j)) \tag{2}$$

Actually, $\mathcal{F}_A^{(t)}(i,j)$ is the weighted sum of all the temporal pixels across T frames. Each temporal pixel $\mathcal{F}^{(n)}(i,j)$ is linearly transformed by W_V, where n is the index referred to T frames in a video. The attention weight $\mathrm{w}^{nt}(i,j)$ indicates the impact from the temporal feature $\mathcal{F}^{(n)}(i,j)$ to $\mathcal{F}^{(t)}(i,j)$, which is calculated as:

$$w^{nt}(i,j) = \frac{\exp\left(g^{nt}(i,j)\right)}{\sum_m \exp\left(g^{mt}(i,j)\right)} \tag{3}$$

$$g^{mt}(i,j) = \frac{(W_Q \mathcal{F}^m(i,j)) \cdot (W_K \mathcal{F}^t(i,j))}{\sqrt{d_k}} \tag{4}$$

W_Q and W_K transform temporal pixels to latent space of d_k dimension linearly. After that, we summarized the whole processing as *Temporal Pixels based Parallel-head Attention(TPPA)* block to compute the temporal context of a video. As shown in Fig.2, this block is established on basic self-attention operation with temporal pixels. An TPPA block capture $H \times W$ context attention features $\{\mathcal{F}_A(i,j)\}_{all\ positions}$ in the same way as Eq.(2). We perform self-attention on each position (i,j) individually and compute them parallelly. Then, the output of the whole computation process is

$$\begin{aligned} \mathcal{F}_{out}^t(i,j) &= \mathcal{F}^t(i,j) + \mathcal{F}_A^t(i,j) \\ where\ i \in [1,H], j \in [1,W], t \in [1,T] \end{aligned} \tag{5}$$

H and W is the spatial size of the feature map, which can be the feature of any stage in CNN. In order to accelerate the multi-way attention computation process, the self-attention module of each set of temporal pixels forms a **parallel-head**, which can be computed in parallel with matrix form on GPU. The block is performed on the entire input sequence of features, which means information propagates across the entire video. On the other hand, we perform self-attention on each spatial position individually and in parallel, which not only captures the temporal context of local regions but also maintains the semantic structure among spatial space.

4 RANDOM FRAMES WITH BOOTSTRAPPING ATTENTION (RFBA)

The TPPA block can model the context of frame sequence well, but how to capture longer range context and how to extract key feature in a long video, are still very important and challenging. Therefore, to capture long-range context by reducing the video redundancy, we are inspired by the Bootstrapping mechanism[9], and designed the **Bootstrapping Attention**, namely, *Random Frames based Bootstrapping Attention* (RFBA), as shown in Fig.1.

4.1 Review of Bootstrapping

Assume that we have data set $\boldsymbol{D}$ and an algorithm $\boldsymbol{A}$, we want to use consensus $\boldsymbol{h_t = A(D_t)}$, $\boldsymbol{(0 \le t < T)}$ to get more stable performance than direct $\boldsymbol{A(D)}$, but we need many more $\boldsymbol{D_t}$ than the $\boldsymbol{D}$ on hand. One solution is approximate math $\boldsymbol{h_t = A(D_t)}$ from $\boldsymbol{D_t \sim Random(D, N)}$ using only $\boldsymbol{D}$ on our hand. $\boldsymbol{Random(D, N)}$ means to randomly select N samples from $\boldsymbol{D}$. In a word, bootstrapping is a statistical tool that re-samples from the whole data set $\boldsymbol{D}$ to 'simulate' more input data set $\boldsymbol{D_t}$, $\boldsymbol{0 \le t < T}$. We re-sample $\boldsymbol{N}$ samples from $\boldsymbol{D}$ randomly with replacement, and obtain h_t by $A(D_t)$, then the final result would be $H = uniform(h_t)$[9]. The bootstrapping is simple, but needs to meet some statistical assumptions. One of our contributions is to verify that the video data satisfies the statistical assumption of using bootstrapping.

4.2 Backbone Network

Our TPPA block and RFBA framework can be combined with any popular DNNs architecture, for example, in this paper, we adopt ResNet and ResNext as our backbone network to build RFBA framework. ResNext is an improved version of ResNet with similar architecture. For a ResNet network, there are 5 stages of convolution layers and 1 classify layer, as shown in Fig.1. Stage 1 consists of a convolution layer and a pooling layer. Stage 2-5 consist of Convolutional Block and Identity Blocks.

4.3 Bootstrapping Attention

Bootstrapping[9] is a strategy for re-sampling, to generate a serious $\boldsymbol{D_t}$, whose distribution is coincident with $\boldsymbol{D}$. With the basic ideas of bootstrapping, each $\boldsymbol{g_t = A(D_t)}$, $\boldsymbol{0 \le t < T}$will be summed up for final decision. For action recognition, $\boldsymbol{D_t}$refers to a set of frames, and after each $\boldsymbol{D_t}$pass the backbone network, the result will be summed up finally. We find that the last step of bootstrapping doesn't consider the diverse impact of different $\boldsymbol{g_t}$. Thus, we introduce the self-attention mechanism to deal with the results of bootstrapping and defined the Random Frames based bootstrapping Attention (RFBA), as shown in Fig.1.

More specifically, as shown in Algorithm.1, the RFBA framework contains three steps. Firstly, with bootstrapping, we sample T frames each time from the input video sequence $\boldsymbol{F}$for M times. Each T frames ($\boldsymbol{F_T^m}$) has a consensus distribution with $\boldsymbol{F}$. Actually, with bootstrapping, the T frames in each $\boldsymbol{F_T^m}$are very sparse according to the value of T, **which also means the video redundancy can be alleviated**. Meanwhile, since we sampled M times and each $\boldsymbol{F_T^m}$has the consensus distribution with $\boldsymbol{F}$, it also guarantees key frames and comprehensive features with enough M.

Secondly, each bootstrapping sample set $\boldsymbol{\{D_m\}_{m=1}^M}$is feed into the convolution network with TPPA block, and the final result of TPPA network is pooling into one feature vector h_m. After that, the M TPPA output attention features $H = \{h^m\}_{m=1}^M$ are feed into another TPPA block, the orange block in Fig.1, for context attention of $\boldsymbol{\{D_m\}_{m=1}^M}$. Formally, it compute the weighted sum of M TPPA

Algorithm 1 The algorithm of RFBA

1: **hyper param** M**:** Sample M times totally
2: **hyper param** T**:** Sample T frames each time
3: Re-sample M times from one video sequence with replacement, obtain M groups input data $D_m, 1 \leq m \leq M$ as in Fig.1
4: **for** each $m \in [1, M]$ **do**
5: Compute the output of TPPA network $h_m = A(F_m)$
6: **end**
7: Compute Bootstrap Attention result H_A using Eq.6
8: Compute the weighted sum $H_A^{'}$ by average pooling
9: **return** The output of RFBA, $H_A^{'}$

features with Eq. 6 as the final feature representation H_A.

$$H_A = softmax(\frac{H \cdot H^T}{scale}) \cdot H \tag{6}$$

5 ACTION RECOGNITION ON STATIC IMAGE

The original intention in designing TPPA and RFBA is to capture longer-range temporal context attention for video-based action recognition. However, to prove the flexible of our approach, we also try to model spatial contextual information with the TPPA block and RFBA framework for action recognition on static images.

In order to successfully apply the proposed approach for static image-based action recognition, the *spatial shift* operation is involved to construct a sequence data at first. Basically, the spatial shift is just to do randomly spatial cropping on static images. These randomly cropped images are combined together as a sequence data, named *pseudo video*. For each image, the spatial shift is implemented K times, and then we can obtain a pseudo video of K frames. Finally, by transforming a still image into such a pseudo video, we can process this pseudo video in the same way as video data for action recognition. The obtained frames can be stacked randomly to a pseudo video because there is no temporal information in the pseudo video and the TPPA process is a temporal-independent process. Actually, the basic spatial shift can be replaced with some other similar operation. For example, Wu et al.[34] presented a parameter-free, FLOP-free shift operation as an alternative to spatial convolutions, which can be used as a more comprehensive spatial shift operation. In fact, the spatial shift operation is simple but effective. We achieve up to 7.44%*mAP* improvement in *Stanford40*[36], which leads us to believe that our method can also capture the long-range spatial context attention well for image-based tasks.

6 EXPERIMENTS

In this section, we conduct extensive experiments and empirical evaluations on two most popular datasets: *UCF101*[25] for videos and *Stanford40*[36] for static images. The whole experiment plan is summarized as follows. In addition, we note that any DNNs with a few TPPA blocks is called *TPPA net*, and the TPPA net with RFBA framework, as Fig.1 shows, is called *RFBA net*, in another word, RFBA net is based on TPPA backbone network. In UCF101 dataset, two different backbone network is adopted: 2D-ResNet and 3D-ResNext. Firstly, to illustrate the effectiveness of TPPA block, 4 experiments are conducted: (1) Comparison of TPPA nets with TPPA block in a different stage of the backbone network. (2) Comparison of TPPA nets with a different number of TPPA block. (3) Comparison of TPPA nets with a different number of input frames. (4) Comparison of TPPA nets with 2D CNN or 3D CNN backbone network. Secondly, to figure out the behavior of the RFBA framework, 3 experiments are conducted: (1) Comparison of single TPPA net and the RFBA net with the backbone network of the TPPA net. (2) Comparison of standard RFBA net and RFBA net without bootstrapping attention (bootstrap model in Tab.4). (3) Comparison of RFBA nets with different hyperparameters M and T.

In Stanford40 dataset, we only use 2D-ResNet as backbone network due to there is no temporal information in the pseudo video. So only 5 ablation experiments are conducted: (1) Comparison of the TPPA net and the basic network without TPPA block. (2) Comparison of TPPA nets with a different number of input frames. (3) Comparison of TPPA nets with a different backbone network. (4) Comparison of RFBA net and single TPPA net. (5) Comparison of RFBA nets with different hyperparameters M and T.

6.1 Data and Performance Metrics

UCF101. *UCF*101[25] has 13320 videos from 101 action categories. We follow the standard setup of *UCF*101 following the practice in [2, 24, 25], and all of the experiments are conducted with the split 1 of official train/test splits. The metrics we use include top-1 and top-5 accuracy, mean average precision (*mAP*).

Stanford40. The dataset contains 40 action classes, and there are 9532 images in total with 180-300 images per action class and training / test ratio is 4000 / 5532. We report the mean average precision (*mAP*) metric to evaluate the performance of our method following the practice in [11, 38].

6.2 Experimental Setup

We conduct all of the experiments using Pytorch framework on 3 Nvidia 1080Ti GPUs. And there are 2 basic DNNs used in our experiments: (i) 2D-ResNet and (ii) 3D-ResNext. We build and train the two basic models following the practice in [24] and [14] separately. The two basic models all obtain reasonable accuracy(about 77.4% and 85.9% top-1 accuracy) compared with the previous results(72.8%[24] and 83.5%[2] top-1 accuracy) with the same architecture.

Training on UCF101. For the training of TPPA net, we conduct experiments of 2D-TPPA Net and 3D-TPPA Net based on basic DNNs. The input of 2D-model is 3 frames with randomly cropping(0.3-1.0), and is resized to 224×224 pixels. We train the 3D-model with the input of 16 frames, each frame is resized 224×224 pixels. All of the TPPA nets in ablation experiments are fine-tuned on basic DNNs, basic DNNs are initialized with ImageNet pre-trained parameters and then trained on UCF101. The learning rate of training is decreased from initial value 0.1 to 0.001 at epoch 300 with the cosine annealing strategy[20]. We fine-tune the model with the fixed learning rate of 0.001.

The RFBA nets are no-trained in our experiments, a simple Attention Module is applied to realize the bootstrapping attention. However, in practice, we also can train the RFBA net end-to-end with a TPPA block as bootstrapping attention, as shown in Fig.1. The TPPA backbone net in RFBA is pre-trained on UCF101, we only

perform inference with RFBA framework and achieve awesome results.

Training on Stanford40. TPPA nets of Stanford40 are all initialized with the pre-trained parameters on ImageNet[4]. The input is the pseudo video with 9 frames when training. Each frame is obtained by shift operation in 5 and is resized to 224×224 pixels. TPPA nets are trained with 2 Nvidia 1080Ti GPU with 30 images in a mini-batch. The maximum iteration size is 68k iterations with AdaGrad[8] optimizer and the fixed learning rate value is 0.0001. We apply the same pre-process as training for inference.

6.3 Experimental Results of Video-based Action Recognition

Model	modality	acc@1
Two-stream[24]	RGB	83.6
Two-stream[24]	RGB + flow	91.2
LSTM[5]	RGB	81.0
3D-fused[2]	RGB	83.2
3D-fused[2]	RGB + flow	89.3
I3D[2]	RGB	84.5
TPPA net[ours]	RGB	**84.8**
RFBA net[ours]	RGB	**91.7**

Table 1: Comparison with state-of-the-art methods on UCF101 split1.

To evaluate the performance of our model, we conducted experiments on the UCF101 split1 for video-based action recognition. As shown in Tab.1, our overall model, i.e. RFBA net, achieved 7.2% top-1 accuracy improvement compared with the methods only based on RGB modality, and also achieve 0.5% top-1 accuracy improvement with multi-modal methods. There are two different backbone network we use in our experiments next, 2D-Resnet101[15] and 3D-Resnext101[14, 35].

To figure out the effect TPPA block, we conduct a serious of experiments on both 2D-resnet backbone and 3D-resnext backbone.

Which stage is the TPPA block in. The TPPA block is added to different stage of ResNet-152 to figure out if the TPPA is effective for 2D ResNet model. As shown in Tab.2, when the TPPA block is added in stage 2, 3, 4, 5 separately, the performance of TPPA nets are

model	N_B	stage	acc@1	acc@5	mAP
2D-Res	0	-	77.380	93.188	77.755
deeperRes	0	-	77.617	93.240	75.494
widerRes	0	-	76.407	92.977	77.957
TPPA net	1	stage2	78.406	93.714	80.757
		stage3	78.301	93.267	78.906
		stage4	78.301	94.161	79.161
		stage5	78.406	93.714	80.757
	5	stage4|5	77.267	93.206	82.368
	10	stage2|3|4|5	**79.509**	**93.526**	**83.885**

Table 2: The TPPA nets with different blocks, where N_B refers to the number of stacked TPPA block.

backbone	model	T	acc@1	acc@5	mAP
$2D-R101$	ResNet	16	77.372	90.960	76.024
	TPPA net	16	80.703	94.819	79.718
		64	83.378	95.642	80.488
		128	**84.768**	**96.669**	81.933
$3D-RX$	ResNext	16	85.938	98.061	89.083
	TPPA net	16	**86.572**	**98.159**	88.799

Table 3: The TPPA nets of different backbone and input, where T refers to the number of input frames

improved compared to the basic 2D-ResNet. And the performance, mAP and top-1 accuracy, are gradually increasing from stage2 case to stage5 case. We argue that TPPA block captures the long-range temporal context to help improve the performance, and due to deeper CNN features have a better representation of the action and smaller spatial size, the TPPA net with TPPA block in the deeper stage will capture the temporal context of the local region better.

How many are the TPPA blocks. To figure out how deep is better for the TPPA network, we compare three models with different number of TPPA blocks inserted to 2D ResNet: one TPPA (1 after stage5), five TPPAs (2 after stage4 and 3 after stage5), ten TPPAs (2 after each stage). Tab.2 shows the results of TPPA nets with different number of TPPA blocks. The results all achieved significant improvement compared to the result of basic 2D-ResNet, 2.1%top-1 accuracy is obtained (79.509 vs 77.380). We argue that more TPPA block can reach better prediction result, but too many blocks also lead to training trouble.

Another question is if the improvement comes from more parameter or deeper layers. To answer the question, we make the basic ResNet-152 deeper (by adding a new FC layer with 2048×2048 parameters before the last fully connected layer) and wider (by replacing the last FC layer to be a new FC layer with 51200×101 parameters). The row 2-3 of Tab.2 shows that a deeper ResNet just obtain small improvement (+0.237% top1 accuracy) but damages about 2.261% on mAP. The difficulty of training more FC layer may lead to the damage of the mAP. Meanwhile, the wider network reduces the top-1 accuracy of 0.973% and the mAP increase 0.202%. In contrast, our approach achieved a better result, +2.1% for top-1 accuracy and +6.1% for mAP, which means that the improvements are not because TPPA blocks introduce more parameters.

More input frames. Tab.3 shows that the TPPA net improve 3.4% top-1 accuracy (80.703 vs 77.372) than basic ResNet-101 with T=16 frames. With the increase of the frames, the accuracy and mAP also increase 7.4% (84.768) and 5.9% (81.933). The results suggest that more input frames benefit the performance of TPPA block.

TPPA net with 3D CNN backbone. The 2D convolution network with TPPA block can capture temporal context. Furthermore, we add the TPPA block to the 3D convolution network in the same way. As shown in Tab. 3, TPPA net with 3D CNN backbone increases the top1 accuracy with about 0.4% (86.572 vs 85.938), which proves that the TPPA and the 3D convolution operation can be complementary in capturing temporal information. This is an awesome attribution that TPPA can be combined with the most popular architectures of action recognition.

The ablation experiments of RFBA nets. To evaluate the effectiveness of the random frames based bootstrapping attention,

backbone	model	T	M	acc@1	acc@5	mAP
$2D-R101$	ResNet	16	-	77.372	90.960	76.024
	TPPAnet	16	-	80.703	94.819	79.718
		3	-	77.161	93.286	74.905
	boostrap	6	4	81.152	94.634	85.455
		3	4	78.536	94.422	83.619
	RFBAnet	16	4	83.378	95.533	86.788
		6	4	81.835	95.208	85.823
		3	4	80.545	94.449	84.769
		3	20	82.294	95.916	87.616
$3D-RX101$	ResNext	16	-	85.938	98.061	89.083
	RFBAnet	16	4	88.820	98.998	91.827
		16	8	**91.742**	**99.167**	**93.501**
		16	10	91.322	99.124	93.154
		16	16	90.284	98.934	92.400

Table 4: Performance of RFBA with M times of bootstrap

we add the RFBA framework on 2D TPPA net and 3D TPPA net separately to measure the effect of RFBA.

For 2D ResNet-101 backbone network, Tab.4 reports the ablation experiments results. There is 2.6% improvement in top-1 accuracy (83.378 vs 80.703) for RFBA net with 16 frames input, compared with single TPPA net. To explore more about RFBA, we change hyperparameters, M and T, of RFBA net for more experimental results. When we have fewer frames T with M=4 times bootstrapping, both the accuracy and mAP decreased, such as top-1 accuracy from 83.378 to 80.545. This infers that the number of frames is a crucial factor because fewer input frames can't represent the distribution of the whole video well. Besides, by changing the bootstrap times M for RFBA net, M from 4 to 20, the top-1 accuracy varies from 80.545 to 82.294, more bootstrapping streams leads to better results.

Actually, RFBA is to combine different bootstrapping streams with the self-attention module, as shown in Fig.1. To emphasize the essence of this self-attention part, we conducted experiments with RFBA framework with or without self-attention module (noted as *bootstrap* in Tab. 4) with $M = 4$ times bootstrap. In Tab. 4, it shows that about 0.7%(81.835 vs 81.152) and 3.2% (80.545 vs 78.536) on top-1 accuracy improvements are obtained separately with our RFBA due to introducing the attention.

For 3D ResNext-101 backbone network, while the default input is 16 frames, we compared the basic 3D ResNext with RFBA net with different bootstrap times. For example, the result of RFBA net with 16 frames each sampling and 4 times bootstrap has about 2.8% (88.82 vs 85.938) improvement on top-1 accuracy. Furthermore, in order to evaluate the bootstrap times effectiveness in RFBA, we conduct experiments with different bootstrap times. The results show that bootstrap more times is not always better, and the best configure for the 3D model with RFBA is $M = 8, T = 16$, which reaches to top-1 accuracy of 91.742%.

By the way, we need to point out that the Bootstrapping attention used here is a **none-trained module**. If we train the RFBA block, we could get better performance. The experiments in Tab.4 also suggest that RFBA can be combined with any existing network.

6.4 Experimental Results of Image-based Action Recognition

The Performance of TPPA net with pseudo video. To prove the TPPA block contributes to the static image-based recognition, we evaluate the TPPA net of ResNet50 backbone with 9 frames pseudo video. The result is shown in 5, there is 5.5% mAP improvement(89.334 vs 83.476) compared with the baseline model. To reduce the influence of the using of pseudo video, we also compare the TPPA net and basic model ResNet with the same input, 9 frames pseudo video. There is still 3.9% on mAP improvement (89.334 vs 85.404). The result infers that the improvement mainly comes from the using of TPPA block. The experiments suggest TPPA block has a strong ability to capture context even with pseudo video.

More pseudo video input frames. To figure out the TPPA net performance with different input, we compare 5 TPPA nets with the same architecture and different input, pseudo videos of 3, 6, 9, 19, 40 frames. The result shows that the 2.2% mAP improvement (89.813 vs 87.688) is obtained when the input is a pseudo video with 40 frames. The results are also consistent with our hypothesis, the more input frames ensure that the attention mechanism captures enough context information.

Deeper TPPA net. We conduct experiments to figure out if there are improvements by deeper backbone network. The result in Tab.5 shows that 0.6%mAP improvement (89.944 vs 89.334) is obtained with the deeper TPPA net. So the experiments next will also be performed with ResNet-101 backbone network.

backbone	model	T	M	mAP
$2D-R50$	ResNet	1	-	83.476
		9	-	85.404
	TPPAnet	3	-	87.688
		6	-	88.785
		9	-	89.334
		19	-	89.615
		40	-	**89.813**
$2D-R101$	TPPA net	3	-	87.439
		9	-	**89.944**
	RFBA net	3	10	90.727
		3	20	**90.916**
		9	3	90.198
		9	10	90.496

Table 5: The ablation experiments of Stanford40

A&A	model	mAP
w/	R*CNN[11]	90.900
	Part action network[38]	91.200
	Part action network(simple version)[38]	84.200
w/o	MOP[12]	74.200
	FV-CNN[3]	75.600
	Absolute + Relative scale coding[16]	80.000
	TPPAnet[ours]	87.439
	RFBAnet[ours]	90.916

Table 6: Comparison with state-of-the-art methods on Stanford40 dataset, *w/ A&A* and *w/o A&A* refer to using *Artificial Annotation* or not when inference.

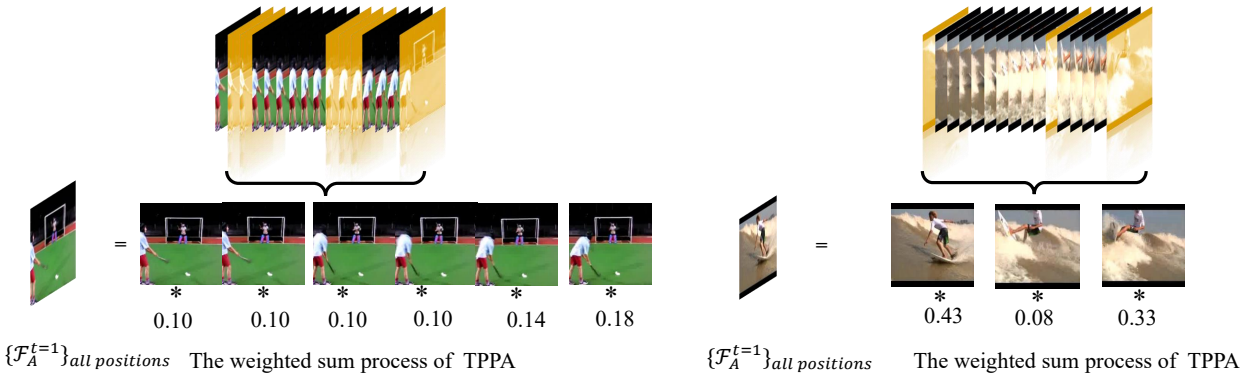

Figure 3: The visualization of the TPPA weighted sum process in UCF101. We show a few frames of the highest weight.

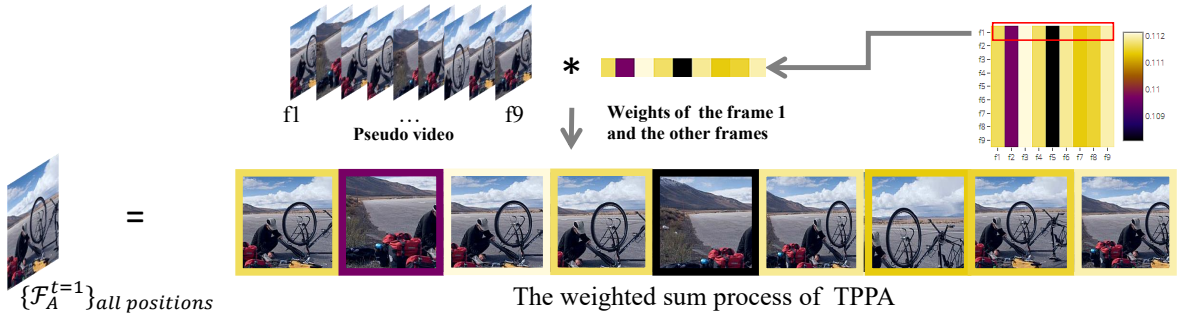

Figure 4: The visualization of the weighted sum process of TPPA block in pseudo video.

The performance of RFBA net with pseudo video. We conduct RFBA nets ablation experiments in Tab.5. Firstly, the results show that all of the RFBA nets achieve improvements compared with TPPA net without RFBA framework, the improvement is up to 3.5%mAP (90.916 vs 87.439). Secondly, we compare RFBA nets with different hyperparameters M and T. With the same input frames T, we perform bootstrapping with more times M, the results are better, +0.3%mAP (90.496 vs 90.198). Finally, The result also suggests that we can achieve good enough result (90.916% on mAP), only with a few input frames (3 frames), by RFBA framework.

The best results on Stanford40. The Tab.6 shows the state-of-the-art results on *stanford*40. Our method achieves 90.916% on mAP with RFBA and TPPA, which is almost the best results on the dataset. We also note that the methods in 1-3 rows use artificial annotation and other algorithms to assist in inference. The part action network[38] depends largely on the performance of Part Affinity Fields Network (PAF)[39] algorithm, not its own effectiveness. It is not fair to compare ours with these methods, but our RFBA net still reaches almost the best result.

7 VISUALIZATION

As mentioned in Section3.2, attention mechanism is a weighted sum process. To figure out the behavior of TPPA block, we visualize the process of UCF101 and Stanford40 separately.

In the visualization of UCF101, we show the attention weighted sum process, i.e. $\{\mathcal{F}_A^{t=1}\}$ using Eq.2. To show the results more clearly, we simplify the visualization in two ways: (i) We only show the computation process of the first frame in each video. (ii) We only show a few frames with the highest weights, i.e. the frames with orange color in Fig.3. In this way, we can find that each local feature (each frame's feature) has the context of the entire video, which is consistent with what we assume.

In the visualization of Stanford40, the weights are related to color, lighter color means higher weight. The *weight matrix* in Fig.4 is the average of the weight matrices of each local region, i.e. $\{w^{nt}\}_{i,j=1,1}^{h,w}$ in Eq.3. In the example of *fixing a bike*. We find that frame 2 and frame 5 have the lowest weights because there is no object of "bike" in frame 2 and frame 5. Obviously, it's extremely hard to recognize the action without the "bike" object. According to the visualization, TPPA will pay more attention to specific local features, so we argue that TPPA can model the context of temporal.

8 CONCLUSION

We proposed an action recognition approach, the Random Frames based Bootstrapping Attention framework. In our method, we also introduced an attention block, i.e. the Temporal Pixels based Parallel-head Attention (TPPA), which can capture the temporal context attention efficiently. Furthermore, we also try to apply this temporal context attention idea to image-based action recognition, by transforming the image into the pseudo video. Finally, our extensive experiments and empirical evaluations show that our approach achieves satisfactory performance.

REFERENCES

[1] Jimmy Ba, Volodymyr Mnih, and Koray Kavukcuoglu. 2014. Multiple object recognition with visual attention. *arXiv preprint arXiv:1412.7755* (2014).

[2] Joao Carreira and Andrew Zisserman. 2017. Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset. (2017).

[3] Mircea Cimpoi, Subhransu Maji, Andrea Vedaldi, Mircea Cimpoi, Subhransu Maji, and Andrea Vedaldi. 2015. Deep filter banks for texture recognition and segmentation. In *Computer Vision Pattern Recognition*.

[4] Jia Deng, Wei Dong, Richard Socher, Li Jia Li, Kai Li, and Fei Fei Li. 2009. ImageNet: a Large-Scale Hierarchical Image Database. In *IEEE Conference on Computer Vision Pattern Recognition*.

[5] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. 2015. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2625–2634.

[6] Tran Du, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. 2015. Learning Spatiotemporal Features with 3D Convolutional Networks. In *IEEE International Conference on Computer Vision*. 4489–4497.

[7] Wenbin Du, Yali Wang, and Qiao Yu. 2017. RPAN: An End-to-End Recurrent Pose-Attention Network for Action Recognition in Videos. In *IEEE International Conference on Computer Vision*. 3725–3734.

[8] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research* 12, 7 (2011), 257–269.

[9] Bradley Efron. 1979. Bootstrap Methods: Another Look at the Jackknife. *Annals of Statistics* 7, 1 (1979), 1–26.

[10] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 580–587.

[11] Georgia Gkioxari, Ross Girshick, and Jitendra Malik. 2015. Contextual Action Recognition with R*CNN. *International Journal of Cancer Journal International Du Cancer* 40, 1 (2015), 1080–1088.

[12] Yunchao Gong, Liwei Wang, Ruiqi Guo, and Svetlana Lazebnik. 2014. Multi-scale Orderless Pooling of Deep Convolutional Activation Features. (2014).

[13] Hu Han, Jiayuan Gu, Zhang Zheng, Jifeng Dai, and Yichen Wei. 2017. Relation Networks for Object Detection. (2017).

[14] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. 2018. Can Spatiotemporal 3D CNNs Retrace the History of 2D CNNs and ImageNet. *computer vision and pattern recognition* (2018), 6546–6555.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. *computer vision and pattern recognition* (2016), 770–778.

[16] Fahad Shahbaz Khan, Van De Weijer Joost, Rao Muhammad Anwer, Andrew D. Bagdanov, Michael Felsberg, and Jorma Laaksonen. 2016. Scale Coding Bag of Deep Features for Human Attribute and Action Recognition. (2016).

[17] Zhenyang Li, Kirill Gavrilyuk, Efstratios Gavves, Mihir Jain, and Cees GM Snoek. 2018. Videolstm convolves, attends and flows for action recognition. *Computer Vision and Image Understanding* 166 (2018), 41–50.

[18] Xinchen Liu, Wu Liu, Huadong Ma, and Huiyuan Fu. 2016. Large-scale vehicle re-identification in urban surveillance videos. In *2016 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 1–6.

[19] Xinchen Liu, Wu Liu, Tao Mei, and Huadong Ma. 2017. Provid: Progressive and multimodal vehicle reidentification for large-scale urban surveillance. *IEEE Transactions on Multimedia* 20, 3 (2017), 645–658.

[20] Ilya Loshchilov and Frank Hutter. 2016. SGDR: Stochastic Gradient Descent with Warm Restarts. (2016).

[21] Huadong Ma, Chengbin Zeng, and Charles X Ling. 2012. A reliable people counting system via multiple cameras. *ACM Transactions on Intelligent Systems and Technology (TIST)* 3, 2 (2012), 31.

[22] Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. 2014. Recurrent models of visual attention. In *Advances in neural information processing systems*. 2204–2212.

[23] Shikhar Sharma, Ryan Kiros, and Ruslan Salakhutdinov. 2015. Action recognition using visual attention. *arXiv preprint arXiv:1511.04119* (2015).

[24] Karen Simonyan and Andrew Zisserman. 2014. Two-Stream Convolutional Networks for Action Recognition in Videos. (2014).

[25] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. 2012. UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild. *Computer Science* (2012).

[26] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. 2015. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*. 4489–4497.

[27] Gül Varol, Ivan Laptev, and Cordelia Schmid. 2018. Long-term temporal convolutions for action recognition. *IEEE transactions on pattern analysis and machine intelligence* 40, 6 (2018), 1510–1517.

[28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. (2017).

[29] Wang, Heng, Liu, Cheng-Lin, Klaeser, Alexander, Schmid, and Cordelia. 2013. Dense Trajectories and Motion Boundary Descriptors for Action;Recognition. *International Journal of Computer Vision* 103, 1 (2013), 60–79.

[30] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. 2016. Temporal segment networks: Towards good practices for deep action recognition. In *European conference on computer vision*. Springer, 20–36.

[31] Limin Wang, Yuanjun Xiong, Wang Zhe, Qiao Yu, Dahua Lin, Xiaoou Tang, and Luc Van Gool. 2016. Temporal Segment Networks: Towards Good Practices for Deep Action Recognition. In *European Conference on Computer Vision*. 20–36.

[32] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. 2017. Non-local Neural Networks. (2017).

[33] Xiaolong Wang and Abhinav Gupta. 2018. Videos as Space-Time Region Graphs. (2018).

[34] Bichen Wu, Alvin Wan, Xiangyu Yue, Peter Jin, Sicheng Zhao, Noah Golmant, Amir Gholaminejad, Joseph Gonzalez, and Kurt Keutzer. 2017. Shift: A Zero FLOP, Zero Parameter Alternative to Spatial Convolutions. (2017).

[35] Saining Xie, Ross Girshick, Piotr Dollar, Zhuowen Tu, and Kaiming He. 2016. Aggregated Residual Transformations for Deep Neural Networks. (2016).

[36] Bangpeng Yao, Xiaoye Jiang, Aditya Khosla, Andy Lai Lin, Leonidas J. Guibas, and Fei Fei Li. 2011. Human action recognition by learning bases of action attributes and parts. In *International Conference on Computer Vision*.

[37] Joe Yue-Hei Ng, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. 2015. Beyond short snippets: Deep networks for video classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4694–4702.

[38] Zhichen Zhao, Huimin Ma, and Shaodi You. 2017. Single image action recognition using semantic body part actions. In *Proceedings of the IEEE International Conference on Computer Vision*. 3391–3399.

[39] Cao Zhe, Tomas Simon, Shih En Wei, and Yaser Sheikh. 2016. Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. (2016).

[40] Bolei Zhou, Alex Andonian, Aude Oliva, and Antonio Torralba. 2018. Temporal relational reasoning in videos. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 803–818.