TAM: Temporal Adaptive Module for Video Recognition

Zhaoyang Liu[1, 2] Limin Wang[1] ✉ Wayne Wu[2] Chen Qian[2] Tong Lu[1]
[1] State Key Lab for Novel Software Technology, Nanjing University, China
[2] SenseTime Research
zyliumy@gmail.com lmwang@nju.edu.cn {wuwenyan,qianchen}@sensetime.com lutong@nju.edu.cn

Abstract

*Video data is with complex temporal dynamics due to various factors such as camera motion, speed variation, and different activities. To effectively capture this diverse motion pattern, this paper presents a new temporal adaptive module (***TAM***) to generate video-specific temporal kernels based on its own feature map. TAM proposes a unique two-level adaptive modeling scheme by decoupling the dynamic kernel into a location sensitive importance map and a location invariant aggregation weight. The importance map is learned in a local temporal window to capture short-term information, while the aggregation weight is generated from a global view with a focus on long-term structure. TAM is a modular block and could be integrated into 2D CNNs to yield a powerful video architecture (TANet) with a very small extra computational cost. The extensive experiments on Kinetics-400 and Something-Something datasets demonstrate that our TAM outperforms other temporal modeling methods consistently, and achieves the state-of-the-art performance under the similar complexity. The code is available at https://github.com/liu-zhy/temporal-adaptive-module.*

1. Introduction

Deep learning has brought great progress for various recognition tasks in image domain, such as image classification [21, 12], object detection [28], and instance segmentation [11]. The key to these successes is to devise flexible and efficient architectures that are capable of learning powerful visual representations from large-scale image datasets [4]. However, deep learning research progress in video understanding is relatively slower, partially due to the high complexity of video data. The core technical problem in video understanding is to design an effective temporal module, that is expected to be able to capture complex temporal structure with high flexibility, while yet to be of low computational consumption for processing high dimensional video data efficiently.

3D Convolutional Neural Networks (3D CNNs) [15, 34] have turned out to be mainstream architectures for video modeling [1, 8, 36, 27]. The 3D convolution is a direct extension over its 2D counterparts and provides a learnable operator for video recognition. However, this simple extension lacks specific consideration about the temporal properties in video data and might as well lead to high computational cost. Therefore, recent methods aim to model video sequences in two different aspects by combining a lightweight temporal module with 2D CNNs to improve efficiency (e.g., TSN [40], TSM [23]), or designing a dedicated temporal module to better capture temporal relation (e.g., Nonlocal Net [41], ARTNet [38], STM [17], TDN [39]). However, how to devise a temporal module with both high efficiency and strong flexibility still remains to be an unsolved problem. Consequently, we aim at advancing the current video architectures along this direction.

In this paper, we focus on devising an adaptive module to capture temporal information in a more flexible way. Intuitvely, we observe that video data is with extremely complex dynamics along the temporal dimension due to factors such as camera motion and various speeds. Thus 3D convolutions (temporal convolutions) might lack enough representation power to describe motion diversity by simply employing a fixed number of **video invariant** kernels. To deal with such complex temporal variations in videos, we argue that **adaptive temporal kernels** for each video are effective and as well necessary to describe motion patterns. To this end, as shown in Figure 1, we present a two-level adaptive modeling scheme to decompose the video specific temporal kernel into a location sensitive *importance map* and a location invariant (also video adaptive) *aggregation kernel*. This unique design allows the location sensitive importance map to focus on enhancing discriminative temporal information from a local view, and enables the video adaptive aggregation to capture temporal dependencies with a global view of the input video sequence.

Specifically, the design of *temporal adaptive module*

✉: Corresponding author.

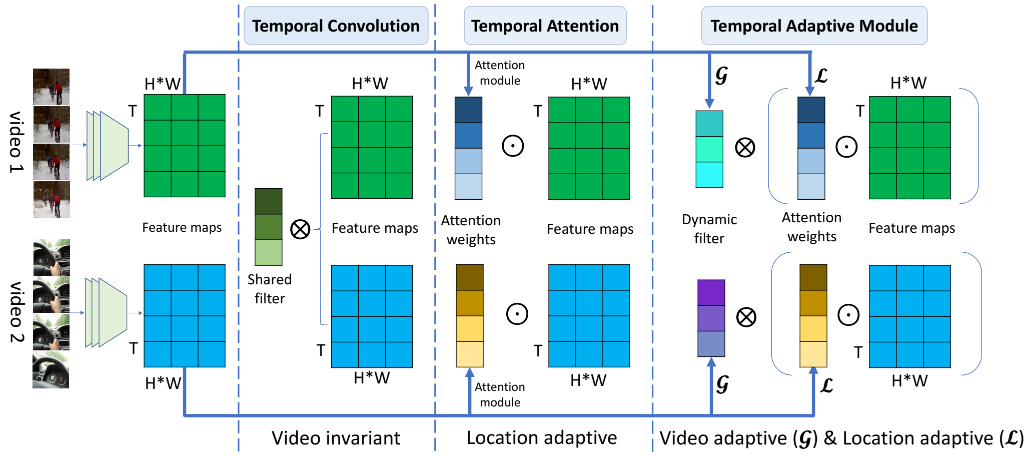

Figure 1. **Temporal module comparisons:** The standard *temporal convolution* shares weights among videos and may lack the flexibility to handle video variations due to the diversity of videos. The *temporal attention* learns position sensitive weights by assigning varied importance for different time without any temporal interaction, and may ignore the long-range temporal dependencies. Our proposed *temporal adaptive module* (TAM) presents a two-level adaptive scheme by learning the local importance weights for location adaptive enhancement and the global kernel weights for video adaptive aggregation. $\odot$ is attention operation, and $\otimes$ is convolution operation.

(TAM) strictly follows two principles: high efficiency and strong flexibility. To ensure our TAM with a low computational cost, we first squeeze the feature map by employing a global spatial pooling, and then establish our TAM in a channel-wise manner to keep the efficiency. Our TAM is composed of two branches: a local branch ($\mathcal{L}$) and a global branch ($\mathcal{G}$). As shown in Fig. 2, TAM is implemented in an efficient way. The local branch employs temporal convolutions to produce the location sensitive importance maps to enhance the local features, while the global branch uses fully connected layers to produce the location invariant kernel for temporal aggregation. The importance map generated by a local temporal window focuses on short-term motion modeling and the aggregation kernel using a global view pays more attention to the long-term temporal information. Furthermore, our TAM could be flexibly plugged into the existing 2D CNNs to yield an efficient video recognition architecture, termed as TANet.

We verify the proposed TANet on the task of action classification in videos. In particular, we first study the performance of the TANet on the Kinetics-400 dataset, and demonstrate that our TAM is better at capturing temporal information than other several counterparts, such as temporal pooling, temporal convolution, TSM [23], TEINet [24], and Non-local block [41]. Our TANet is able to yield a very competitive accuracy with the FLOPs similar to 2D CNNs. We further test our TANet on the motion dominated dataset of Something-Something, where the state-of-the-art performance is achieved.

2. Related Work

Video understanding is a core topic in the field of computer vision. At the early stage, a lot of traditional methods [22, 20, 29, 43] have designed various hand-crafted features to encode the video data, but these methods are too inflexible when generalized to other video tasks. Recently, since the rapid development of video understanding has been much benefited from deep learning methods [21, 32, 12], especially in video recognition, a series of CNNs-based methods were proposed to learn spatiotemporal representation, and the differences with our method will be clarified later. Furthermore, our work also relates to dynamic convolution and attention in CNNs.

CNNs-based methods for action recognition. Since the deep learning method has been wildly used in the image tasks, there are many attempts [18, 31, 40, 46, 10, 23, 39] based on 2D CNNs devoted to modeling the video clips. In particular, [40] used the frames sparsely sampled from the whole video to learn the long-range information by aggregating scores after the last fully-connected layer. [23] shifted the channels along the temporal dimension in an efficient way, which yields a good performance with 2D CNNs. By a simple extension from spatial domain to spatiotemporal domain, 3D convolution [15, 34] was proposed to capture the motion information encoded in video clips. Due to the release of large-scale Kinetics dataset [19], 3D CNNs [1] were wildly used in action recognition. Its variants [27, 36, 44] decomposed the 3D convolution into a spatial 2D convolution and a temporal 1D convolution to learn the spatiotemporal features. And [8] designed a network with dual paths to learn the spatiotemporal features and achieved a promising accuracy in video understanding.

The methods aforementioned all share a common insight that they are video invariant and ignore the inherent temporal diversities in videos. As opposed to these methods, we design a two-level adaptive modeling scheme by decompos-

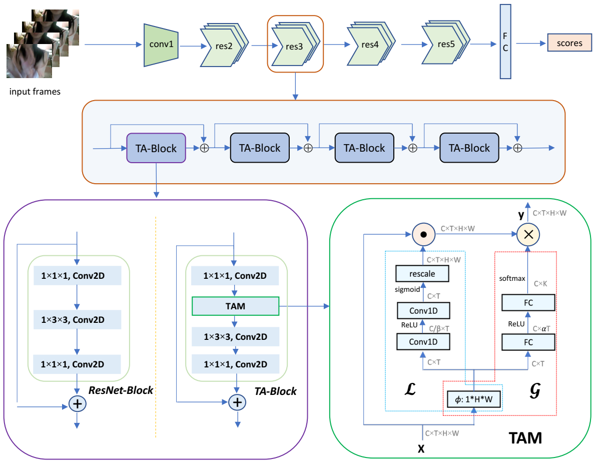

Figure 2. **The overall architecture of TANet:** ResNet-Block *vs.* TA-Block. The whole workflow of temporal adaptive module (TAM) in the lower right shows how it works. The shape of tensor has noted after each step. $\oplus$ denotes element-wise addition, $\odot$ is element-wise multiplication, and $\otimes$ is convolution operation. The symbols appeared in figure will be explained in Sec. 3.1.

ing the video specific operation into a location sensitive excitation and a location invariant convolution with adaptive kernel for each video clip.

Attention in action recognition. The local branch in TAM mostly relates to SENet [13]. But the SENet learned modulation weights for each channel of feature maps. Several methods [24, 5] also resorted to the attention to learn more discriminative features in videos. Different from these methods, the local branch keeps the temporal information to learn the location sensitive importances. [41] designed a non-local block which can be seen as self-attention to capture long-range dependencies. Our TANet captures the long-range dependencies by simply stacking more TAM, and keep the efficiency of networks.

Dynamic convolutions. [16] first proposed the dynamic filters on the tasks of video and stereo prediction, and designed a convolutional encoder-decoder as filter-generating network. Several works [45, 3] in image tasks attempted to generate aggregation weights for a set of convolutional kernels, and then produce a dynamic kernel. Our motivation is different from these methods. We aim to use this temporal adaptive module to deal with temporal variations in videos. Specifically, we design an efficient form to implement this temporal dynamic kernel based on input feature maps, which is critical for understanding the video content.

3. Method

3.1. The Overview of Temporal Adaptive Module

As we discussed in Sec.1, video data typically exhibit the complex temporal dynamics caused by many factors such as camera motion and speed variations. Therefore, we aim to tackle this issue by introducing a temporal adaptive module (TAM) with video specific kernels, unlike the sharing convolutional kernel in 3D CNNs. Our TAM could be easily integrated into the existing 2D CNNs (e.g., ResNet) to yield a video network architecture, as shown in Figure 2. We will give an overview of TAM and then describe its technical details.

Formally, let $X \in \mathbb{R}^{C\times T\times H\times W}$ denote the feature maps for a video clip, where C represents the number of channels, and T, H, W are its spatiotemporal dimensions. For efficiency, TAM only focuses on temporal modeling and the spatial pattern is expected to captured by 2D convolutions. Therefore, we first employ a global spatial average pooling to squeeze the feature map as follows:

$$\hat{X}_{c,t} = \phi(X)_{c,t} = \frac{1}{H\times W}\sum_{i,j} X_{c,t,j,i}, \qquad (1)$$

where c, t, j, i is the index of different dimensions (in channel, time, height and width), and $\hat{X} \in \mathbb{R}^{C\times T}$ aggregates the spatial information of X. For simplicity, we here use ϕ to denote the function that aggregates the spatial information. The proposed temporal adaptive module (TAM) is established based on this squeezed 1D temporal signal with high efficiency.

Our TAM is composed of two branches: a local branch $\mathcal{L}$ and a global branch $\mathcal{G}$, which aims to learn a location sensitive importance map to enhance discriminative features and then produces the location invariant weights to adaptively aggregate temporal information in a convolutional manner. More specifically, the TAM is formulated as follows:

$$Y = \mathcal{G}(X) \otimes (\mathcal{L}(X) \odot X), \qquad (2)$$

where $\otimes$ denotes convolution operation and $\odot$ is element-wise multiplication. It is worth noting that these two branches focus on different aspects of temporal information, where the local branch tries to capture the short term information to attend important features by using a temporal convolution, while the global branch aims to incorporate long-range temporal structure to guide adaptive temporal aggregation with fully connected layers. Disentangling kernel learning procedures into local and global branches turns out to be an effective way in experiments. These two branches will be introduced in the following sections.

3.2. Local Branch in TAM

As discussed above, the local branch is location sensitive and aims to leverage short-term temporal dynamics to perform video specific operation. Given that the short-term information varies slowly along the temporal dimension, it is thus required to learn a location sensitive importance map to discriminate the local temporal semantics.

As shown in Figure 2, the local branch is built by a sequence of temporal convolutional layers with *ReLU* non-linearity. Since the goal of local branch is to capture short term information, we set the kernel size K as 3 to learn importance map solely based on a local temporal window. To control the model complexity, the first $\mathrm{Conv1D}$ followed by BN [14] reduces the number of channels from C to $\frac{C}{\beta}$. Then, the second $\mathrm{Conv1D}$ with a sigmoid activation yields the importance weights $V \in \mathbb{R}^{C\times T}$ which are sensitive to temporal location. Finally, the temporal excitation is formulated as follows:

$$Z = \mathrm{F}_{\text{rescale}}(V) \odot X = \mathcal{L}(X) \odot X, \tag{3}$$

where $\odot$ denotes the element-wise multiplication and $Z \in \mathbb{R}^{C\times T\times H\times W}$. To match size of X, $\mathrm{F}_{\text{rescale}}(V)$ rescales the V to $\hat{V} \in \mathbb{R}^{C\times T\times H\times W}$ by replicating in spatial dimension.

3.3. Global Branch in TAM

The global branch is location invariant and focuses on generating an adaptive kernel based on long-term temporal information. It incorporates global context information and learns to produce the location invariant and also video adaptive convolution kernel for dynamic aggregation.

Learning the Adaptive Kernels. We here opt to generate the dynamic kernel for each video clip and aggregate temporal information in a convolutional manner. To simply this procedure and as well as preserve high efficiency, The adaptive convolution will be applied in a channel-wise manner. In this sense, the learned adaptive kernel is expected to only model the temporal relations without taking channel correlation into account. Thus, our TAM would not change the number of channels of input feature maps, and the learned adaptive kernel convolves the input feature maps in a channel-wise manner. More formally, for the c^{th} channel, the adaptive kernel is learned as follows:

$$\Theta_c = \mathcal{G}(X)_c = \mathrm{softmax}(\mathcal{F}(\mathbf{W_2}, \delta(\mathcal{F}(\mathbf{W_1}, \phi(X)_c)))), \tag{4}$$

where $\Theta_c \in \mathcal{R}^K$ is generated adaptive kernel (aggregation weights) for c^{th} channel, K is the adaptive kernel size, δ denotes the activation function *ReLU*. The adaptive kernel is also learned based on the squeezed feature map $\hat{X}_c \in \mathbb{R}^T$ without taking the spatial structure into account for modeling efficiency. But different with the local branch, we use fully connected (fc) layers $\mathcal{F}$ to learn the adaptive kernel by leveraging long-term information. The learned adaptive kernel with the global receptive field, thus could aggregate temporal features guided by the global context. To increase the modeling capabilities of the global branch, we stack two fc layers and the learned kernel is normalized with a softmax function to yield a positive aggregation weight. The learned aggregation weights $\Theta = \{\Theta_1, \Theta_2, ..., \Theta_C\}$ will be employed to perform video adaptive convolution.

Temporal Adaptive Aggregation. Before introducing the adaptive aggregation, we can look back on how a vanilla temporal convolution aggregates the spatio-temporal visual information:

$$Y = W \otimes X, \tag{5}$$

Where W is the weights of convolution kernel and has no concern with input video samples in inference. We argue this fashion ignores the temporal dynamics in videos, and thus propose a video adaptive aggregation:

$$Y = \mathcal{G}(X) \otimes X, \tag{6}$$

where $\mathcal{G}$ can be seen as a kernel generator function. The kernel generated by $\mathcal{G}$ can perform adaptive convolution but is shared cross temporal dimension and still location invariant. To address this issue, the local branch produces Z with location sensitive importance map. The whole procedures can be expressed as follows:

$$Y_{c,t,j,i} = \mathcal{G}(X) \otimes Z = \Theta \otimes Z = \sum_k \Theta_{c,k} \cdot Z_{c,t+k,j,i}, \tag{7}$$

where $\cdot$ denotes the scalar multiplication and Y is the output feature maps ($Y \in \mathbb{R}^{C\times T\times H\times W}$).

In summary, TAM presents an adaptive module with a unique aggregation scheme, where the location sensitive excitation and location invariant aggregation all derive from input features, but focus on capturing different structures (*i.e.*, short-term and long-term temporal structure).

3.4. Exemplar: TANet

We here intend to describe how to instantiate the TANet. Temporal adaptive module can endow the existing 2D CNNs with a strong ability to model different temporal structures in video clips. In practice, TAM only causes limited computing overhead, but obviously improves the performance on different types of datasets.

ResNets [12] are employed as backbones to verify the effectiveness of TAM. As illustrated in Fig. 2, the TAM is embedded into ResNet-Block after the first Conv2D, which easily turns the vanilla ResNet-Block into TA-Block. This fashion will not excessively alter the topology of networks and can reuse the weights of ResNet-Block. Supposing we sample T frames as an input clip, the scores of T frames after fc will be aggregated by average pooling to yield the clip-level scores. No temporal downsampling is performed before fc layer. The extensive experiments are conducted in Sec. 4 to demonstrate the flexibility and efficacy of TANet.

Discussions. We notice that the structure of local branch is similar to the SENet [13] and STC [5]. The first obvious difference is the local branch does not squeeze the temporal dimension. We thus use temporal 1D convolution, instead of fc layer, as a basic layer. Two-layer design only seeks to make a trade-off between non-linear fitting capability and model complexity. The local branch provides the location sensitive information, and thereby addresses the issue that the global branch is insensitive to temporal location.

TSN [40] and TSM [23] only aggregate the temporal features with a fixed scheme, but TAM can yield the video specific weights to adaptively aggregate the temporal features in different stages. In extreme cases, our global branch in TAM can degenerate into TSN when dynamic kernel weights Θ is learned to equal to $[0, 1, 0]$. From another perspective, if the kernel weights Θ is set to $[1, 0, 0]$ or $[0, 0, 1]$, global branch can be turned into TSM. It seems that our TAM theoretically provides a more general and flexible form to model the video data.

When it refers to 3D convolution [15], all input samples share the same convolution kernel without being aware of the temporal diversities in videos as well. In addition, our global branch essentially performs a video adaptive convolution whose filter has size $1 \times k \times 1 \times 1$, while each filter in a normal 3D convolution has size $C \times k \times k \times k$, where C is the number of channels and k denotes the receptive field. Thus our method is more efficient than 3D CNNs. Unlike some current dynamic convolution [3, 45], TAM is more flexible, and can directly generate the kernel weights to perform video adaptive convolution.

4. Experiments

4.1. Datasets

Our experiments are conducted on three large scale datasets, namely, Kinetics-400 [19] and Something-Something (Sth-Sth) V1&V2 [9]. Kinetics-400 contains ∼300k video clips with 400 human action categories. The trimmed videos in Kinetics-400 are around 10s. We train the models on the training set (∼240k video clips), and test models on the validation set (∼20k video clips). The Sth-Sth datasets focus on fine-grained and motion-dominated action, which contains pre-defined basic actions involving different interacting objects. The Sth-Sth V1 comprises ∼86k video clips in the training set and ∼12k video clips in the validation set. Sth-Sth V2 is an updated version of Sth-Sth V1, which contains ∼169k video clips in the training set and ∼25k video clips in the validation set. They both have 174 action categories.

4.2. Implementation Details

Training. In our experiments, we train the models with 8 and 16 frames as inputs. On Kinetics-400, following the practice in [41], the frames are sampled from 64 consecutive frames in the video. On Sth-Sth V1&V2, the uniform sampling strategy in TSN [40] is employed to train TANet. We first resize the shorter side of frames to 256, and apply the multi-scale cropping and randomly horizontal flipping as data augmentation. The cropped frames are resized to 224×224 for network training. The batch size is 64. Our models are initialized by ImageNet pre-trained weights to reduce the training time. Specifically, on the Kinetics-400, the epoch for training is 100. The initial learning rate is set 0.01 and divided by 10 at 50, 75, 90 epochs. We use SGD with a momentum of 0.9 and a weight decay of 1e-4 to train TANet. On Sth-Sth V1&V2, we train models with 50 epochs. The learning rate starts at 0.01 and is divided by 10 at 30, 40, 45 epoch. We use a momentum of 0.9 and a weight decay of 1e-3 to reduce the risk of overfitting.

Testing. Different inference schemes are applied to fairly compare with other state-of-the-art models. On kinetics-400, we resize the shorter to 256 and take 3 crops of 256×256 to cover the spatial dimensions. In the temporal dimension, we uniformly sample 10 clips for 8-frame models and 4 clips for 16-frame models. The final video-level prediction is yielded by averaging the scores of all spatio-temporal views. On Sth-Sth V1, we scale the shorter side of frames to 256 and use center crop of 224×224 for evaluation. On Sth-Sth V2, we employ a similar evaluation protocol to Kinetics, but only uniformly sample 2 clips, and also present the accuracy with a single clip using center crop.

4.3. Ablation Studies

The exploration studies are performed on Kinetics-400 to investigate different aspects of TANet. The ResNet architecture we used is the same with [12]. Our TANet replaces all ResNet-Blocks with TA-Blocks by default.

Parameter choices. We use different combinations of α and β to figure out the optimal hyper-parameters in TAM. The TANet is instantiated as in Fig. 2. TANet with $\alpha = 2$ and $\beta = 4$ achieves the highest performance shown in Table 1a, which will be applied in following experiments.

Temporal receptive fields. We try to increase the temporal receptive fields for learned kernel Θ in the global branch. From Table 1b, it seems the larger K is beneficial to the accuracy when TANet takes more sampled frames as inputs. On the other hand, it even degenerates the performance of TANet when sampling 8 frames. In our following experiments, the K will be set to 3.

TAM in the different position. Table 1c tries to study the effects of TAM in different position. TANet-a, TANet-b, TANet-c, and TANet-d denote the TAM is inserted before the first convolution, after the first convolution, after the second convolution, and after the last convolution in the block, respectively. These four styles are graphically presented in the *supplementary material*. The style in Fig. 2 is TANet-b, which has a slightly better performance than other styles as shown in Table 1c. The TANet-b will be abbreviated as TANet by default in the following experiments.

Setting		Frames	Top-1	Top-5
α=1	β=4	8	75.63%	92.10%
α=2	β=4	8	**76.28%**	**92.60%**
α=4	β=4	8	75.72%	92.14%
α=2	β=2	8	75.91%	92.38%
α=2	β=4	8	**76.28%**	**92.60%**
α=2	β=8	8	75.63%	92.20%

(a) **Study on parameter choices of α and β.** The results have revealed an excellent advantage that TAM is not so insensitive to these hyper-parameters.

Kernel	Frames	Top-1	Top-5
K=3	8	76.28%	92.60%
K=5	8	75.62%	92.14%
K=3	16	76.87%	92.88%
K=5	16	77.19%	93.17%

(b) **Study on the temporal receptive fields**. Trying larger temporal receptive fields of Θ when TANet uses 8 frames and 16 frames as inputs.

Model	Frames	Top-1	Top-5
TANet-a	8	75.95%	92.18%
TANet-b	8	**76.28%**	**92.60%**
TANet-c	8	75.75%	92.13%
TANet-d	8	75.20%	91.78%

(c) **Exploring where to insert TAM.** we insert TAM into TA-Block in difference position to study its impact.

Stages	Frames	Blocks	Top-1	Top-5
res_5	8	3	74.12%	91.45%
res_{4-5}	8	9	75.15%	92.04%
res_{3-5}	8	13	75.90%	92.22%
res_{2-5}	8	16	**76.28%**	**92.60%**

(d) **Study on the number of TA-Blocks**. The TA-Blocks are cumulatively added into ResNet50 from stage2 to stage5.

Models	ShuffleNet V2		MobileNet V2		Inception V3		ResNet-50		I3D-ResNet-50 [2]	
	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
w/o TAM	62.1%	84.3%	64.1%	85.6%	71.4%	89.8%	70.2%	88.9%	76.6%	-
with TAM	67.3%	87.6%	71.6%	90.1%	75.6%	92.0%	76.3%	92.6%	77.2%	92.9%
Δ*Acc.*	***+ 5.2%***	***+ 3.3%***	***+ 7.5%***	***+ 4.5%***	***+ 4.2%***	***+ 2.2%***	***+ 6.1%***	***+ 3.7%***	***+ 0.6%***	**-**

(e) **Study on the impact on different backbones**. We try to extend the TAM to other backbones. I3D-ResNet-50 takes 32 frame as inputs but other backbones take 8 frame as inputs. The performance shows TAM can easily enjoy the benefits with different backbones.

Table 1. **Ablation studies on Kinetics-400**. These experiments use ResNet-50 as backbone and take 8-frame as inputs in training. All models share the same inference protocol, i.e., 10 clips × 3 crops.

The number of TA-Blocks. To make a trade-off between performance and efficiency, we gradually add more TA-Blocks into ResNet. As shown in Table 1d, we find that more TA-Blocks contributes to better performance. The res_{2-5} achieves the highest performance and will be used in our experiments.

Transferring to other backbones. Finally we verify the generalization of our proposed module. To this end, we apply the TAM to other well known 2D backbones, like ShuffleNet V2 [26], MobileNet V2 [30], Inception V3 [33] and 3D backbones, like I3D-ResNet-50 [1, 2], where all models has no temporal downsampling operation before the global average pooling layer. From Table 1e, we can observe that the backbone networks equipped with our TAM outperform their C2D and I3D baselines by a large margin, which demonstrates the generalization ability of our proposed module.

Models	FLOPs (of single view)	Params	Top-1	Top-5
C2D	42.95G	24.33M	70.2%	88.9%
C2D-Pool	42.95G	24.33M	73.1%	90.6%
C2D-TConv	53.02G	28.10M	73.3%	90.7%
C2D-TIM [24]	43.06G	24.37M	74.7%	91.7%
$\text{I3D}_{3\times1\times1}$	62.55G	32.99M	74.3%	91.6%
TSM* [23]	42.95G	24.33M	74.1%	91.2%
TEINet* [24]	43.01G	25.11M	74.9%	91.8%
NL C2D [41]	64.49G	31.69M	74.4%	91.5%
Global branch	43.00G	24.33M	75.6%	91.9%
Local branch	43.07G	25.59M	73.3%	90.7%
Global branch + SE [13]	43.02G	24.65M	75.9%	92.1%
TANet-R	43.02G	25.59M	76.0%	92.2%
TANet	43.02G	25.59M	**76.3**%	**92.6**%

Table 2. **Studying on the effectiveness of TAM.** All models use ResNet50 as backbone and take 8 frames with sampling stride 8 as inputs. To be consistent with testing, the FLOPs are calculated with spatial size 256 × 256. $\star$ is reported by the author of paper. All methods share the same training setting and inference protocol.

4.4. Comparison with Other Temporal Modules

As a standard temporal operator, we make comparisons between our TAM and other temporal modules. For fair comparison, all models in this study employ the same frame input (8×8) and backbone (ResNet-50). The inference protocol is to sample 10×3 crops to report the performance.

Baselines. We first choose several baselines with temporal modules. We begin with the **2D ConvNet (C2D)**, where we only build 2D ConvNet with ResNet50 and focus on learning the spatial features. In this sense, it operates on each frame independently without any temporal interaction before the global average pooling layer in the end. The second is the **C2D-Pool.** To endow the 2D network with temporal modeling capacity, C2D-Pool inserts the average pooling layer whose kernel size is $K \times 1 \times 1$ to perform temporal fusion without any temporal downsampling. This is easily implemented by simply replacing all TAMs in network with average pooling layers. The third type is the learnable temporal convolution, whose kernel is shared by all videos. We first replace each TAM with a standard temporal convolution with randomly initialized weights, termed as **C2D-TConv**. In addition, we replace the standard temporal convolution with the channel-wised temporal convolution using TSM [23] initialization to solely aggregate temporal information without relating different channels, termed as **C2D+TIM** [24]. Finally, we compare with **Inflated 3D ConvNet (I3D)**, whose operation is also based on temporal convolutions by directly inflating the original 2D convolutions into 3D convolutions. In our implementation, we inflate the first 1×1 kernel in ResNet-Block to $3 \times 1 \times 1$, which can provide a more fair comparison with our TANet.

Following [41], this variant is referred to as I3D$_{3\times1\times1}$. It is worth noting these three types of temporal convolutions share the similar idea of fixed aggregation kernel, but differ in the specific implementation details, which can demonstrate the efficacy of adaptive aggregation in our TAM.

The aforementioned methods share the same temporal modeling scheme with a fixed pooling or convolution. As shown in Table 2, our TAM yields superior performance to all of them. We observe that C2D obtains the worst performance that is less than TAM by 6.1%. Surprisingly, the naively-implemented temporal convolution (C2D-TConv) performs similar to temporal pooling (C2D-Pool) (73.3% vs. 73.1%), which can partly blame on the randomly initialized weights of temporal convolution that corrupt the ImageNet pre-trained weights. In temporal convolution based models, we find that C2D-TIM obtains the best performance with the smallest number of FLOPs. We analyze that this channel-wise temporal convolution can well keep the feature channel correspondence and thus benefits most from the ImageNet pre-trained models. However, it is still worse than our TAM by 1.6%.

Other temporal counterparts. There are some competitive temporal modules that learn video features based on C2D, i.e., **TSM** [23], **TEINet** [24], and **Non-local C2D (NL C2D)**. We here compare our TAM with these different temporal modules, and the results of TSM and TEINet are directly cited from the original papers, as they share similar numbers of FLOPs to our TAM. The non-local block is a kind of self-attention module, proposed to capture the long-range dependencies in videos. The preferable setting with 5 non-local blocks mentioned in [41] is under a similar computational budget and thereby employed to compare with our TAM. As seen in Table 2, our TANet achieves highest accuracy among these temporal modules, outperforming TSM by 2.2%, TEINet by 1.4%, and NL C2D by 1.9%.

Variants of TAM. To study the performance of each part in temporal adaptive module, we separately validate the **Global branch** and **Local branch**. Furthermore, **Global branch + SE** uses global branch with SE module [13] to compare with TANet. TANet achieves the highest accuracy among these models as well, which proves the efficacy of each part of TAM and as well as the strong complementarity between local branch and global branch. We also reverse the order of local branch and global branch (TANet-R): $Y = \mathcal{L}(X) \odot (\mathcal{G}(X) \otimes X)$. We see that TANet is slightly better than TANet-R.

4.5. Comparison with the State of the Art

Comparison on Kinetics-400. Table 3 shows the state-of-the-art results on Kinetics-400. Our method (TANet) achieves the competitive performance to other models. TANet-50 with 8-frame also outperforms SlowFast [8] by 0.7% when using similar FLOPs per view. The 16-frame TANet only uses 4 clips and 3 crops for evaluation such that it provides higher inference efficiency and more fair comparisons with other models. It is worth noting that our 16-frame TANet-50 is still more accurate than 32-frame NL I3D by 1.4%. As ip-CSN [35] is pretrained on Sports-1M [18], it achieves the promising accuracy with deeper backbone, i.e., ResNet152. Furthermore, TAM is compatible with the existing video frameworks like SlowFast. Specifically, our TAM is more lightweight than a standard $3 \times 1 \times 1$ convolution when taking the same number of frames as inputs, but can yield a better performance. TAM thus can easily replace the $3\times1\times1$ convolution in SlowFast to achieve lower computational costs. X3D has achieved great success in video recognition. X3D was searched by massive computing resources and can not be easily extended in a new situation. Although our method fails to beat all state-of-the-art methods with deeper networks, TAM as a lightweight operator can enjoy the advantages from more powerful backbones and video frameworks. To sum up, the proposed TANet makes a good practice on adaptively modeling the temporal relations in videos.

Methods	Backbones	Training Input	GFLOPs	Top-1	Top-5
TSN [40]	InceptionV3	3×224×224	3×250	72.5%	90.2%
ARTNet [38]	ResNet18	16 ×112×112	24×250	70.7%	89.3%
I3D [1]	InceptionV1	64×224×224	108×N/A	72.1%	90.3%
R(2+1)D [36]	ResNet34	32×112×112	152×10	74.3%	91.4%
NL I3D [41]	ResNet50	128×224×224	282×30	76.5%	92.6%
ip-CSN [35]	ResNet50	8×224×224	1.2×10	70.8%	-
TSM [23]	ResNet50	16×224×224	65×30	74.7%	91.4%
TEINet [24]	ResNet50	16×224×224	86×30	76.2%	92.5%
bLVNet-TAM [6]	bLResNet50	48×224×224	93×9	73.5%	91.2%
SlowOnly [8]	ResNet50	8×224×224	42×30	74.8%	91.6%
SlowFast$_{4\times16}$ [8]	ResNet50	(4+32)×224×224	36×30	75.6%	92.1%
SlowFast$_{8\times8}$ [8]	ResNet50	(8+32)×224×224	66×30	77.0%	92.6%
I3D* [2]	ResNet50	32 ×224×224	335 ×30	76.6%	-
TANet-50	ResNet50	8×224×224	43×30	76.3%	92.6%
TANet-50	ResNet50	16×224×224	86×12	76.9%	92.9%
X3D-XL [7]	-	16×312×312	48×30	79.1%	93.9%
CorrNet [37]	ResNet101	32×10×3	224×30	79.2%	-
ip-CSN [35]	ResNet152	32 ×224×224	83×30	79.2%	93.8%
SlowFast$_{16\times8}$ [8]	ResNet101	(16+64)×224×224	213×30	78.9%	93.5%
TANet-101	ResNet101	8×224×224	82×30	77.1%	93.1%
TANet-101	ResNet101	16×224×224	164×12	78.4%	93.5%
TANet-152	ResNet152	16×224×224	242×12	**79.3%**	**94.1%**

Table 3. Comparisons with the state-of-the-art methods on Kinetics-400. As described in [8], the GFLOPs of a single view × the number of views (temporal clips with spatial crops) represents the model complexity. The GFLOPs is calculated with spatial size 256×256. * denotes the I3D without temporal downsampling.

Comparison on Sth-Sth V1 & V2. As shown in Table 4, our method achieves the comparable accuracy comparing with other models on Sth-Sth V1. For fair comparison, Table 4 only reports the results taking a single clip with a center crop as inputs. TANet$_{En}$ is higher than TSM$_{En}$ equipped with same backbone (Top-1: 50.6% vs. Top-1: 49.7%). We also conduct the experiments on Sth-Sth V2. V2 has more video clips than V1, which can further unleash

Methods	Backbones	Pre-train	Frames	FLOPs	Top-1	Top-5
TSN-RGB [40]	BNInception	ImgNet	$8f$	16G	19.5%	-
TRN-Multiscale [46]	BNInception	ImgNet	$8f$	33G	34.4%	-
S3D-G [44]	Inception	ImgNet	$64f$	71.38G	48.2%	78.7%
ECO [47]	BNIncep+Res18	K400	$16f$	64G	41.6%	-
ECO$_{En}$Lite [47]	BNIncep+Res18	K400	$92f$	267G	46.4%	-
TSN [40]	ResNet50	ImgNet	$8f$	33G	19.7%	46.6%
I3D [42]	ResNet50	ImgNet+K400	$32f \times 2$	306G	41.6%	72.2%
NL I3D [42]	ResNet50	ImgNet+K400	$32f \times 2$	334G	44.4%	76.0%
NL I3D+GCN [42]	ResNet50+GCN	ImgNet+K400	$32f \times 2$	606G	46.1%	76.8%
TSM [23]	ResNet50	ImgNet	$8f$	33G	45.6%	74.2%
TSM [23]	ResNet50	ImgNet	$16f$	65G	47.2%	77.1%
TSM$_{En}$ [23]	ResNet50	ImgNet	$8f$+$16f$	98G	49.7%	78.5%
TAM [6]	ResNet50	ImgNet	$8f$	-	46.1%	-
bLVNet-TAM [6]	ResNet50	Sth-Sth V1	$32f$	48G	48.4%	78.8%
GST [25]	ResNet50	ImgNet	$8f$	30G	47.0%	76.1%
GST [25]	ResNet50	ImgNet	$16f$	59G	48.6%	77.9%
TEINet [24]	ResNet50	ImgNet	$8f$	33G	47.4%	-
TEINet [24]	ResNet50	ImgNet	$16f$	66G	49.9%	-
TEINet$_{En}$ [24]	ResNet50	ImgNet	$8f$+$16f$	66G	52.5%	-
TANet	ResNet50	ImgNet	$8f$	33G	47.3%	75.8%
TANet	ResNet50	ImgNet	$16f$	66G	47.6%	77.7%
TANet$_{En}$	ResNet50	ImgNet	$8f$+$16f$	99G	50.6%	79.3%

Table 4. Comparisons with the state-of-the-art methods on Sth-Sth V1. The models only taking RGB frames as inputs are listed in table. To be consistent with testing, we use spatial size 224×224 to compute the FLOPs.

Methods	Backbones	Pre-train	Frames×clips×crops	Top-1	Top-5
TRN [46]	BNInception	ImgNet	$8f$×2×3	48.8%	77.6%
TSM [23]	ResNet50	ImgNet	$8f$×2×3	59.1%	85.6%
TSM [23]	ResNet50	ImgNet	$16f$×2×3	63.4%	88.5%
TSM$_{2stream}$ [23]	ResNet50	ImgNet	($16f$+$16f$)×2×3	66.0%	90.5%
GST [25]	ResNet50	ImgNet	$8f$×1×1	61.6%	87.2%
GST [25]	ResNet50	ImgNet	$16f$×1×1	62.6%	87.9%
bLVNet-TAM [6]	ResNet50	Sth-Sth V2	$32f$×1×1	61.7%	88.1%
TEINet [24]	ResNet50	ImgNet	$8f$×1×1	61.3%	-%
TEINet [24]	ResNet50	ImgNet	$16f$×1×1	62.1%	-%
TEINet$_{En}$ [24]	ResNet50	ImgNet	($8f$+$16f$)×10×3	66.5%	-%
TANet	ResNet50	ImgNet	$8f$×1×1	60.5%	86.2%
TANet	ResNet50	ImgNet	$8f$×2×3	62.7%	88.0%
TANet	ResNet50	ImgNet	$16f$×1×1	62.5%	87.6%
TANet	ResNet50	ImgNet	$16f$×2×3	64.6%	89.5%
TANet$_{En}$	ResNet50	ImgNet	($8f$+$16f$)×2×3	66.0%	90.1%

Table 5. Comparisons with the SOTA on Sth-Sth V2. We here apply the two different inference protocal, *i.e.*, 1 clip × 1 crop and 2 clip × 3 crop, to fairly evaluate the TAM with other methods.

the full capabilities of TANet without suffering the overfitting. Following the common practice in [23], TANets use 2 clips with 3 crops to evaluate the accuracy. As shown in Table 5, our models have achieved the state-of-art performance on Sth-Sth V2. As a result, the TANet$_{En}$ yields a competitive accuracy compared with the two-stream TSM and TEINet$_{En}$. The results on Sth-Sth V1 & V2 have demonstrated that our method is also good at modeling the fine-grained and motion-dominated actions.

4.6. Visualization of Learned Kernels

To better understand the behavior of TANet, we visualize the distribution of kernel Θ generated by global branch in the last block of stage4 and stage5. For clear comparison, the kernel weights in I3D$_{3\times1\times1}$ at the same stages are also visualized to find more insights. As depicted in Fig. 3, we find that the learned kernel Θ has a different property: the shapes and scales of distribution are more diverse than I3D$_{3\times1\times1}$. Since all video clips share the same kernels in I3D$_{3\times1\times1}$, it causes the kernel weights cluster together tightly. As opposed to temporal convolution, even modeling the same action in different videos, TAM can generate the kernel with slightly different distributions. Taking driving car as an example, the shapes of the distribution shown in Fig. 3 are similar to each other but the medians of distributions are not equal. For different actions like drinking beer and skydiving, the shapes and medians of distributions are greatly different. Even for different videos of the same action, TAM can learn a different distribution of kernel weights. Concerning that the motion in different videos may exhibit different patterns, it is necessary to employ an adaptive scheme to model video sequences.

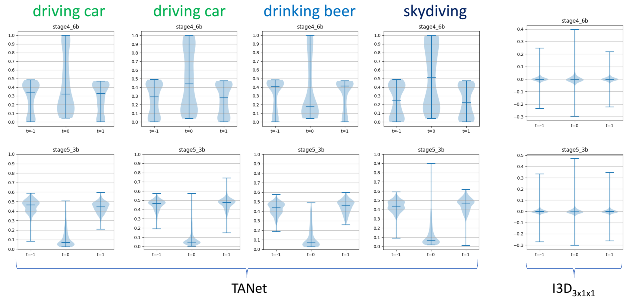

Figure 3. The statistics of kernel weights trained on Kinetics-400, and we plots the distributions in different temporal offsets ($t \in \{-1, 0, 1\}$). Each filled area in violinplot represents the entire data range, which marks the minimum, the median and the maximum. The first four columns in the left figure are the distributions of learned kernels in TANet. In the fifth column, we visualize the filters of $3 \times 1 \times 1$ kernel in I3D$_{3\times1\times1}$ to compare with the TANet.

5. Conclusion

In this paper, we have presented a generic temporal module, termed as temporal adaptive module (TAM), to capture complex motion patterns in videos and proposed a powerful video architecture (TANet) based on this new temporal module. TAM is able to yield a video-specific kernel with the combination of a local importance map and a global aggregation kernel. This unique design is helpful to capture the complex temporal structure in videos and contributes to more effective and robust temporal modeling. As demonstrated on the Kinetics-400, the networks equipped with TAM are better than the existing temporal modules in action recognition, which demonstrates the efficacy of our TAM in video temporal modeling. TANet also achieves the state-of-the-art performance on the motion dominated datasets of Sth-Sth V1&V2.

Acknowledgements. Thanks to Zhan Tong, Jintao Lin and Yue Zhao for the help. This work is supported by National Natural Science Foundation of China (No. 62076119, No. 61921006), SenseTime Research Fund for Young Scholars, Program for Innovative Talents and Entrepreneur in Jiangsu Province, and Collaborative Innovation Center of Novel Software Technology.

References

[1] João Carreira and Andrew Zisserman. Quo vadis, action recognition? A new model and the kinetics dataset. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 4724–4733, 2017.

[2] Chun-Fu Richard Chen, Rameswar Panda, Kandan Ramakrishnan, Rogerio Feris, John Cohn, Aude Oliva, and Quanfu Fan. Deep analysis of cnn-based spatio-temporal representations for action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6165–6175, 2021.

[3] Yinpeng Chen, Xiyang Dai, Mengchen Liu, Dongdong Chen, Lu Yuan, and Zicheng Liu. Dynamic convolution: Attention over convolution kernels. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 11027–11036. IEEE, 2020.

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. Imagenet: A large-scale hierarchical image database. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition CVPR*, pages 248–255, 2009.

[5] Ali Diba, Mohsen Fayyaz, Vivek Sharma, Mohammad Mahdi Arzani, Rahman Yousefzadeh, Juergen Gall, and Luc Van Gool. Spatio-temporal channel correlation networks for action classification. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision - ECCV*, volume 11208 of *Lecture Notes in Computer Science*, pages 299–315, 2018.

[6] Quanfu Fan, Chun-Fu (Richard) Chen, Hilde Kuehne, Marco Pistoia, and David D. Cox. More is less: Learning efficient video representations by big-little network and depthwise temporal aggregation. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada*, pages 2261–2270, 2019.

[7] Christoph Feichtenhofer. X3D: expanding architectures for efficient video recognition. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 200–210. IEEE, 2020.

[8] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *IEEE/CVF International Conference on Computer Vision, ICCV*, pages 6201–6210, 2019.

[9] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fründ, Peter Yianilos, Moritz Mueller-Freitag, Florian Hoppe, Christian Thurau, Ingo Bax, and Roland Memisevic. The "something something" video database for learning and evaluating visual common sense. In *IEEE International Conference on Computer Vision, ICCV*, pages 5843–5851, 2017.

[10] Dongliang He, Zhichao Zhou, Chuang Gan, Fu Li, Xiao Liu, Yandong Li, Limin Wang, and Shilei Wen. Stnet: Local and global spatial-temporal modeling for action recognition. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI*, pages 8401–8408, 2019.

[11] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. In *IEEE International Conference on Computer Vision, ICCV*, pages 2980–2988. IEEE Computer Society, 2017.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 770–778, 2016.

[13] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 7132–7141, 2018.

[14] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis R. Bach and David M. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 448–456. JMLR.org, 2015.

[15] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. In Johannes Fürnkranz and Thorsten Joachims, editors, *Proceedings of the 27th International Conference on Machine Learning (ICML)*, pages 495–502, 2010.

[16] Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc Van Gool. Dynamic filter networks. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 667–675, 2016.

[17] Boyuan Jiang, Mengmeng Wang, Weihao Gan, Wei Wu, and Junjie Yan. STM: spatiotemporal and motion encoding for action recognition. *CoRR*, abs/1908.02486, 2019.

[18] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Fei-Fei Li. Large-scale video classification with convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 1725–1732, 2014.

[19] Will Kay, João Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset. *CoRR*, abs/1705.06950, 2017.

[20] Alexander Kläser, Marcin Marszalek, and Cordelia Schmid. A spatio-temporal descriptor based on 3d-gradients. In Mark Everingham, Chris J. Needham, and Roberto Fraile, editors, *Proceedings of the British Machine Vision Conference 2008*, pages 1–10. British Machine Vision Association, 2008.

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou, and Kilian Q. Weinberger,

editors, *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012.*, pages 1106–1114, 2012.

[22] Quoc V. Le, Will Y. Zou, Serena Y. Yeung, and Andrew Y. Ng. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 3361–3368. IEEE Computer Society, 2011.

[23] Ji Lin, Chuang Gan, and Song Han. TSM: temporal shift module for efficient video understanding. In *IEEE International Conference on Computer Vision, ICCV 2019*, pages 7082–7092, 2019.

[24] Zhaoyang Liu, Donghao Luo, Yabiao Wang, Limin Wang, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Tong Lu. Teinet: Towards an efficient architecture for video recognition. *CoRR*, abs/1911.09435, 2019.

[25] Chenxu Luo and Alan L. Yuille. Grouped spatial-temporal aggregation for efficient action recognition. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 5511–5520. IEEE, 2019.

[26] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet V2: practical guidelines for efficient CNN architecture design. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XIV*, volume 11218 of *Lecture Notes in Computer Science*, pages 122–138. Springer, 2018.

[27] Zhaofan Qiu, Ting Yao, and Tao Mei. Learning spatio-temporal representation with pseudo-3d residual networks. In *IEEE International Conference on Computer Vision, ICCV*, pages 5534–5542, 2017.

[28] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(6):1137–1149, 2017.

[29] Sreemanananth Sadanand and Jason J. Corso. Action bank: A high-level representation of activity in video. In *2012 IEEE Conference on Computer Vision and Pattern Recognition, 2012*, pages 1234–1241. IEEE Computer Society, 2012.

[30] Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 4510–4520. IEEE Computer Society, 2018.

[31] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014*, pages 568–576, 2014.

[32] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015*, 2015.

[33] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2818–2826. IEEE Computer Society, 2016.

[34] Du Tran, Lubomir D. Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *IEEE International Conference on Computer Vision, ICCV*, pages 4489–4497, 2015.

[35] Du Tran, Heng Wang, Matt Feiszli, and Lorenzo Torresani. Video classification with channel-separated convolutional networks. In *IEEE International Conference on Computer Vision, ICCV 2019*, pages 5551–5560, 2019.

[36] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 6450–6459, 2018.

[37] Heng Wang, Du Tran, Lorenzo Torresani, and Matt Feiszli. Video modeling with correlation networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 352–361, 2020.

[38] Limin Wang, Wei Li, Wen Li, and Luc Van Gool. Appearance-and-relation networks for video classification. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 1430–1439, 2018.

[39] Limin Wang, Zhan Tong, Bin Ji, and Gangshan Wu. Tdn: Temporal difference networks for efficient action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1895–1904, 2021.

[40] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *Computer Vision - ECCV*, pages 20–36, 2016.

[41] Xiaolong Wang, Ross B. Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 7794–7803, 2018.

[42] Xiaolong Wang and Abhinav Gupta. Videos as space-time region graphs. In *Computer Vision - ECCV*, pages 413–431, 2018.

[43] Geert Willems, Tinne Tuytelaars, and Luc Van Gool. An efficient dense and scale-invariant spatio-temporal interest point detector. In David A. Forsyth, Philip H. S. Torr, and Andrew Zisserman, editors, *Computer Vision - ECCV*, volume 5303 of *Lecture Notes in Computer Science*, pages 650–663, 2008.

[44] Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *Computer Vision - ECCV*, pages 318–335, 2018.

[45] Brandon Yang, Gabriel Bender, Quoc V. Le, and Jiquan Ngiam. Condconv: Conditionally parameterized convolutions for efficient inference. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Ad-*

vances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada, pages 1305–1316, 2019.

[46] Bolei Zhou, Alex Andonian, Aude Oliva, and Antonio Torralba. Temporal relational reasoning in videos. In *Computer Vision - ECCV*, pages 831–846, 2018.

[47] Mohammadreza Zolfaghari, Kamaljeet Singh, and Thomas Brox. ECO: efficient convolutional network for online video understanding. In *Computer Vision - ECCV*, pages 713–730, 2018.