Motion-driven Visual Tempo Learning for Video-based Action Recognition

Yuanzhong Liu, Junsong Yuan, *Fellow, IEEE,* and Zhigang Tu, *Member, IEEE*

***Abstract*—Action visual tempo characterizes the dynamics and the temporal scale of an action, which is helpful to distinguish human actions that share high similarities in visual dynamics and appearance. Previous methods capture the visual tempo either by sampling raw videos with multiple rates, which require a costly multi-layer network to handle each rate, or by hierarchically sampling backbone features, which rely heavily on high-level features that miss fine-grained temporal dynamics. In this work, we propose a Temporal Correlation Module (TCM), which can be easily embedded into the current action recognition backbones in a plug-in-and-play manner, to extract action visual tempo from low-level backbone features at single-layer remarkably. Specifically, our TCM contains two main components: a Multi-scale Temporal Dynamics Module (MTDM) and a Temporal Attention Module (TAM). MTDM applies a correlation operation to learn pixel-wise fine-grained temporal dynamics for both fast-tempo and slow-tempo. TAM adaptively emphasizes expressive features and suppresses inessential ones via analyzing the global information across various tempos. Extensive experiments conducted on several action recognition benchmarks, e.g. Something-Something V1 & V2, Kinetics-400, UCF-101, and HMDB-51, have demonstrated that the proposed TCM is effective to promote the performance of the existing video-based action recognition models for a large margin. The source code is publicly released at https://github.com/yzfly/TCM.**

***Index Terms*—Action Recognition, Visual Tempo, Multi-scale Temporal Structure, Temporal Correlation Module.**

I. Introduction

DUE to the success of applying deep learning methods on video understanding tasks, the accuracy of video action recognition has improved significantly over the past years [1], [2], [3], [4], [5]. However, modeling action visual tempo in videos is often overlooked. Action visual tempo describes how fast an action goes, which tends to determine the time duration at the temporal scale for recognition [6]. Different person performs the action at his/her own action visual tempo due to various factors e.g. age, gender, strength, and mood, etc. The complexity of action visual tempo leads to a large difference in the temporal dynamics and temporal scale. Failing to capture the action visual tempo in videos may hinder to improve the accuracy of action recognition, especially in some cases where the human actions have high similarity in dynamics and appearance (e.g., walking, jogging, and running), as distinguishing them heavily depend on extracting their action visual tempo information.

Yuanzhong Liu and Zhigang Tu are with the State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, 430079 Wuhan, China. (E-Mail: tuzhigang@whu.edu.cn, yzliu.me@whu.edu.cn). Yuanzhong Liu and Zhigang Tu contributed equally.

Junsong Yuan is with the Computer Science and Engineering department, State University of New York at Buffalo, USA. (Email: jsyuan@buffalo.edu)

Recently, a few attempts [6], [7], [8], [9] have been proposed to address this issue. SlowFast [8] samples video frames at two different rates as input to form a two-pathway SlowFast model for video recognition, where the *slow* pathway operates at a low frame rate while the *fast* pathway operates at a high frame rate. The backbone subnetworks accordingly aggregate the fast-tempo and slow-tempo information jointly and then handle action instances at two temporal scales. Noticeable improvements have been obtained, but this method remains computationally expensive to process action visual tempo since it uses different frame sampling rates. Inspired by the feature-level pyramid networks [10], [11], [12] which can deal with large variance in spatial scales, TPN [6] constructs a temporal pyramid by collecting backbone features from multi-layers and aggregating them to capture the action visual tempo information at the feature-level. TPN shows consistent improvement on several action recognition datasets, but it extremely relies on the temporal modeling ability of the backbone network itself, limiting to gain the benefits from low-level features for action recognition.

Although the high-level features contain more semantic information, useful fine-grained temporal dynamics and temporal scale information are not fully utilized in TPN. Low-level features contain abundant fine-grained temporal dynamics and temporal scale information, thus it is not wise to sacrifice the modeling of low-level action visual tempo for improving the performance. Neglecting the process of low-level features will also result in the loss of semantic information. On the other side, high-level features are considered to contain more action semantic information since they have a large theoretical receptive field formed by deep stacks of convolutional operations [13]. However, in fact, studies [14] have found that the effective receptive field in CNN only takes a small portion of the theoretical receptive field. Large effective receptive field is vital for capturing long-distance dependencies [13], [15]. Consequently, to enlarge the effective receptive field, it is necessary to perform the multi-scale process for low-level features directly.

Motivated by the optical flow estimation methods [16], [17], [18] and the motion representation-based action recognition methods [19], [20], [21], which estimate fine-grained motion information from low-level features to facilitate video analysis, we propose a Temporal Correlation Module (TCM) to capture the action visual tempo from the low-level features at multi-scale temporal dimension to promote the performance of current video-based action recognition models [22], [23], [24]. As shown in Fig. 1, TCM is composed of two parts: a Multi-scale Temporal Dynamics Module (MTDM) which is used

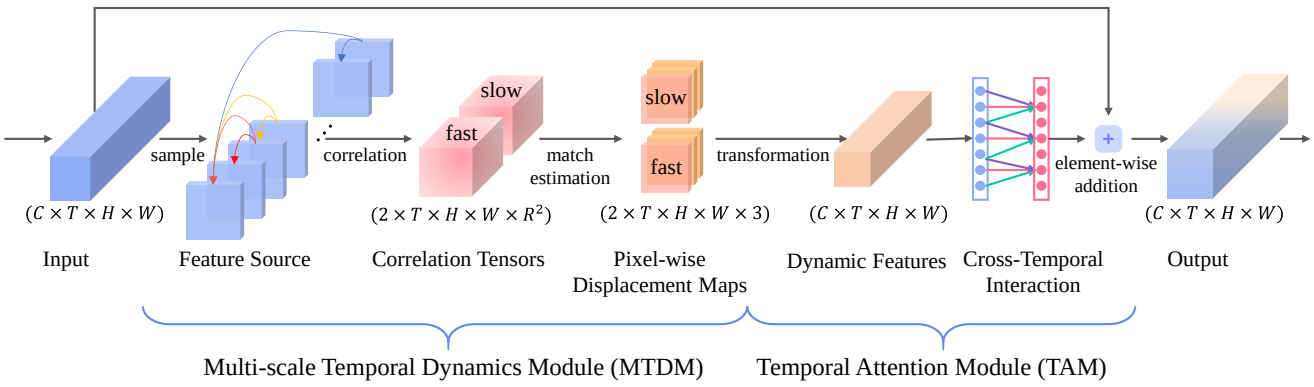


Fig. 1. The architecture of our TCM. TCM includes two main components: MTDM and TAM. In MTDM, given a single-layer backbone features as *Input*, we utilize a multi-scale sampling strategy to *sample* the longest scale and shortest scale feature pairs for each frame to capture the *Feature Source*. A *correlation* is applied to both the longest scale feature pairs and the shortest scale feature pairs to form two *Correlation Tensors* respectively for slow-tempo and fast-tempo. Then, we perform a *motion estimation* to extract the *Pixel-wise Displacement Maps* as MTDM's output, which will be fed into TAM. TAM applies *transformation* on the *Pixel-wise Displacement Map* to exploit *Dynamic Features*. After that, a *Cross-temporal Interaction* is executed to learn temporal attention weights for useful action visual tempo features excitation. Finally, the obtained action visual tempo features are combined with the *Input* features as the *Output*. $C; T; H; W$ respectively represents the features' channel, temporal dimension, height, and width.

for extracting both the slow-tempo and fast-tempo temporal dynamics, and a Temporal Attention Module (TAM) which is used for aggregating the temporal dynamics. Specifically, in MTDM, the low-level backbone features (e.g. the output features of layer *res2* and layer *res3* in the backbone shown in Table I) will serve as the feature source to establish the shortest and longest temporal feature pairs for each video frame. Then, we apply the correlation operation [25] to each feature pairs to construct a correlation tensor, which is processed by an efficient motion estimation method [20], to extract pixel-wise fine-grained temporal dynamics for both fast-tempo and slow-tempo at each frame. The output of MTDM will be fed into the downstream TAM for enhancement. TAM can adaptively highlight discriminate features and reduce insignificant ones by taking the interaction of temporal dynamics at different scales into account.

Correspondingly, by equipping with the explored TCM, a powerful neural network – TCM-Net is constructed. We integrate our TCM into the low-level layer of various action recognition backbone networks and evaluate it extensively on the popular action recognition benchmark datasets: Kinetics-400 [26], HMDB-51 [27], UCF-101 [28], and Something-Something V1 & V2 [29]. The experimental results show that the prior action recognition methods can achieve impressive gains when combine with our TCM. As pointed in [30], [31], [32], most of human actions cannot be recognized in the temporal dominated videos without considering the temporal relationship, like the human actions in the Something-Something V1 & V2 video datasets. Specifically, when incorporating TCM into the basic backbone ResNet50 [33] (TCM is placed right behind layer `res3`, see Table I for layer reference), the modified TCM-R50 model (with only 4% more FLOPs than ResNet50) produces competitive result, which is on par with the prior best performance on the Something-Something V1 & V2 datasets and the Kinetics400 dataset. Besides, a comprehensive ablation studies also demonstrate the effectiveness and efficiency of the two components of TCM *i.e.* MTDM and TAM.

Our main contributions are summarized as follows:

We design a MTDM to fully extract the pixel-wise fine-grained temporal dynamics of both fast-tempo and slow-tempo from the low-level single-layer deep features, which addresses the limitations of the previous methods that heavily rely on high-level features and are unable to exploit benefits from low-level features.

We exploit a TAM, which can adaptively select and enhance the most effective action visual tempo from multi-scale temporal dynamics, to aggregate temporal dynamics.

A TCM is constructed by combining MTDM with TAM, which can incorporate with various action recognition backbone networks easily in a plug-in-and-play way. Extensive experiments conducted on the main 2D and 3D action recognition backbones and action recognition benchmarks show that our TCM significantly improves the accuracy of current video-based action recognition models.

II. Related Work

A. Action Recognition in Videos

The current convolutional deep learning methods [34], [23], [35] dedicated to human action recognition can be roughly divided into two categories, *i.e.* 3D convolutional networks (3D CNNs) and 2D convolutional networks (2D CNNs). 3D CNNs [36], [37], [38], [4] utilize 3D convolutional kernels to jointly model temporal and spatial semantics. Local temporal convolution operations are stacked to capture the long-range temporal dynamics. The Non-local network [15] introduces a non-local operation to better exploit the long-range temporal dynamics from input sequences. Except the non-local operation, there are many other modifications [39], [40], [19]

have been explored to 3D CNNs to boost its performance, but the variation of action visual tempo is often neglected. 2D CNNs [41], [42], [32], [43] apply 2D kernels over per-frame inputs to exploit spatial semantics and followed by a module to aggregate temporal dynamics. The most famous 2D CNN model is the two-steam network [1], [41], [44], in which one stream extracts the RGB appearance features, and the other stream learns the optical flow motion information. Finally, it uses the average pooling for spatio-temporal aggregation. A number of efforts [30], [45], [31], [20] has been carried out to enhance the temporal information extraction efficiency for 2D CNNs. STM [30] proposes a Channel-wise Spatiotemporal Module and a Channel-wise Motion Module to encode the complementary spatiotemporal and motion features in a unified 2D CNN framework. ActionS-ST-VLAD [45] propose a novel action-stage(ActionS) emphasized spatiotemporal vector of locally aggregated descriptors (ActionS-ST-VLAD) method to adaptively aggregate video-level informative deep features. TEA [31] calculates the feature-level temporal differences from spatiotemporal features and utilizes the differences to excite the motion-sensitive channels of the features. MotionSqueeze [20] presents a trainable neural module to establish correspondence across frames and convert them into motion features. These methods provide fine-grained modeling ability to learn adjacent frame temporal dynamics, but they ignore the importance of action visual tempo.

B. Action visual tempo Modeling in Video

Many methods [6], [7], [8] are dedicated to action visual tempo [46] dynamics modeling by taking advantages of the input-level frame pyramid. DTPN [7] samples video frames with varied frame sampling rates and constructs a pyramidal feature representation for arbitrary-length input videos, where the slow-tempo and fast-tempo temporal dynamics can be captured. Such sample strategy tends to require multiple frames which causes a heavy computational cost, especially when the frame sampling rate increases. SlowFast [8] uses a two-level frame pyramid to hard-code the variance of the action visual tempo. Branches are carefully devised to separately process each level, and the mid-level features of these branches are fused interactively. SlowFast can robustly deal with the variance of the action visual tempo, but the multi-branch network is costly. TPN [6] leverages different depth features hierarchically formed inside the backbone network to model action visual tempo. TPN can be applied to various models and brings consistent improvement, but it is limited by the backbone networks' temporal modeling ability. It cannot take advantage of the low-level features, and the useful long-range fine-grained temporal dynamics from distant frames in high-level features may be weakened as it is obtained by stacking multiple local temporal convolutions. To overcome these disadvantages, we utilize the correlation operation to establish pixel-wise matching values for different temporal-scale features and exploit the action visual tempo in videos. We also explore a temporal attention module for interactive action visual tempo feature fusion, which not only enhances the saliency temporal scales, but also enriches the temporal dynamics.

III. Proposed Method

In this section, we will introduce the details of our proposed TCM (see Fig. 1). TCM contains two important parts: a Multi-scale Temporal Dynamics Module (MTDM) and a Temporal Attention Module (TAM). Initially, the visual contents of the input video are encoded into a feature sequence by a spatio-temporal action recognition backbone network. Then, the designed MTDM utilizes a correlation operation to extract the temporal dynamics of both fast-tempo and slow-tempo from this feature sequence. To determine the most effective visual tempo information, TAM will adaptively emphasize expressive temporal features and suppress insignificant ones by analyzing across-temporal interactions. Lastly, the obtained action visual tempo features are combined with the appearance features for final prediction.

A. Multi-scale Temporal Dynamics Module (MTDM)

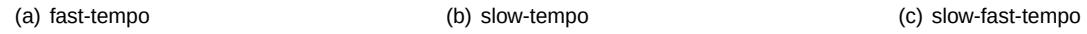
The MTDM is a learnable temporal dynamics extractor, which extracts effective temporal dynamic features of both the fast-tempo and the slow-tempo in three steps: feature source utilization, visual similarity computation, and motion estimation.

1) Feature Source Utilization: The prior feature-based methods [6] utilize the high-level backbone features to construct a multi-layer feature pyramid that has increasing temporal receptive fields from bottom to top. The single-layer pyramid feature source has also been explored [6], but the effectiveness is limited due to the constraint of the backbone's temporal modeling ability. In contrast, we design a novel approach to extract the action visual tempo features of both the slow-tempo and the fast-tempo, where they are obtained by sampling deep features at different rates. This approach can effectively use the single-layer deep features and be free from the constraints of the backbone network.

Specifically, to learn features of the fast-tempo at each frame, we present an adjacent video frames feature extraction strategy (see Fig. 2(a)), which can effectively exploit the shortest scale temporal information. On the other side, to capture features of the slow-tempo for each moment, we use the longest scale temporal information and accordingly construct the feature extraction strategy as shown in Fig. 2(b). Combining the fast-tempo (Fig. 2(a)) with the slow-tempo (Fig. 2(b)) feature sampling strategy, our final multi-scale slow-fast-tempo sampling strategy (Fig. 2(c)) is formed. Accordingly, for each frame, we have its shortest and longest temporal range information, *i.e.* the fast-tempo and the slow-tempo. Inspired by the way to estimate optical flow [47], [48], we calculate the pixel-wise visual similarity characteristics of each frame temporally to model the action visual tempo structure.

2) Visual Similarity Computation: Let us denote the pair of the input feature maps at a certain interval r by $F^t \in \mathbb{R}^{C \times H \times W}$ and $F^{t+r} \in \mathbb{R}^{C \times H \times W}$, where $C; H$ and W are respectively the channel dimension, height and width. The visual similarity score at a position $\mathbf{x}$ with respect to the displacement $\mathbf{p}$ can be defined as:

$$S(\mathbf{x}; \mathbf{p}; t) = F^t_{\mathbf{x}} \quad F^{t+r}_{\mathbf{x}+\mathbf{p}}; \tag{1}$$



(a) fast-tempo (b) slow-tempo (c) slow-fast-tempo

Fig. 2. Slow-fast-tempo Feature Sources. The longest scale and the shortest scale are considered for each moment to form a multi-scale sampling strategy to model the slow-fast-tempo information. The longest scale has a large temporal receptive field for slow-tempo, and the shortest scale has a small temporal receptive field for fast-tempo.

Fig. 3. Our explored Temporal Attention Module (TAM). The displacement map (with confidence map) from MTDM are transformed by six convolution layers to interpret slow-fast tempo semantics. Then the global average pooling (GAP) is used to capture aggregated features. With consideration of the cross-temporal interaction, temporal weights are generated by performing a fast 1D temporal convolution with, where K is adaptively determined by Eq. 8, here we show the case of $K = 3$.

where $\cdot$ represents dot product. To improve the efficiency, we compute the visual similarity score at the position only in its neighborhood with the radius R i.e. $p \in [x - R, x + R]^2$.

Furthermore, to form the correlation tensor, we calculate the visual similarity score of the input feature map pairs F^t and F^{t+r} at each position, which can be computed as following:

$$C(F^t, F^{t+r}) \in \mathbb{R}^{H \times W \times R \times R},$$
$$C_{ijkl}(i, j, k - i + R, l - j + R) = \sum_h F^t_{hij} \cdot F^{t+r}_{hkl} \quad (2)$$

The radius R is set manually and affects the final performance. In practice, given a feature map with the spatial resolution $H \times W$ (H is generally equal to W), we can set $R = INT(H/2)$, where INT is the integer ceiling function. In theory, the result produced by the correlation is four-dimensional ($H \times W \times R \times R$), we resize it to $H \times W \times R^2$ to facilitate subsequent processing.

3) Motion Estimation: To establish the correspondence across frame pairs, we use a light-weight method as MotionSqueeze [20] to estimate optical flow in terms of the correlation tensor $C(F^t, F^{t+r})$. Following the setting of MotionSqueeze [20], we compute optical flow and it's corresponding confidence map for motion information extraction. We find that the confidence map is very useful for identifying motion outliers and learning informative motion features. The two-channels' optical flow and the one-channel' confidence map are concatenated to form a pixel-wise displacement map. To maintain the temporal dimension consistency, we simply duplicate the last temporal dimension of the displacement map. The displacement map contains motion information of specific scale. After performing motion estimation of the longest and shortest scales, we concatenate them together for downstream feature transformation of both the slow-tempo and the fast-tempo.

B. Temporal Attention Module (TAM)

Our designed TAM aims to exploit the temporal dynamics of the upstream input, and adaptively highlight the distinctive features while suppressing the trivial ones by taking across-temporal dynamics interaction into account. It is known that the depth-wise separable convolution [49] can significantly reduce the computational complexity of CNNs, and we use it to improve the efficiency here. As shown in Fig. 3, the displacement map (with the confidence map) from upstream layers will be fed into six convolution layers for transformation. Three $1 \times 3 \times 3$ layers are utilized to approximate a $1 \times 7 \times 7$ layer, then followed by another three $1 \times 3 \times 3$ layers. Specifically, except for the first two layers, all the other layers are followed by a 1×1 convolution layer. The semantics of the displacement map and the confidence map are

expected to be interpreted by the feature transformation. After transforming the displacement map, the slow-fast visual tempo features are obtained for aggregation. The TAM is designed to automatically extrude the discriminative action visual tempo features meanwhile to reduce the impact of inessential features during training. Recently, there are some effective attempts to enhance the temporal information by utilizing the attention mechanism. TEA [31] employs a global average pooling layer to summarize the spatial information to get attentive weights to stimulate the motion-sensitive channels. Motion pattern has been excited and enhanced, but processing the temporal channels in isolation can lead to the loss of cross-temporal interaction.

Given the temporal aggregated feature $F \in R^T$, where T denotes the feature temporal dimension, the temporal attention can be learned without dimensionality reduction according to Equation 3:

$$w = \sigma(W F^T); \tag{3}$$

where W is a general parameter matrix with T elements. Specifically, the parameter matrix in TEA [31] is computed according to Equation 4:

$$W_1 = \begin{bmatrix} w^{1,1} & & 0 \\ \vdots & \ddots & \vdots \\ 0 & & w^{T,T} \end{bmatrix}; \tag{4}$$

where W_1 is a diagonal matrix containing T parameters, but the cross-temporal interaction is completely ignored here. Recent research about attention mechanism [50] suggests that the cross-channel interaction is useful, and the temporal interaction has a latent important function for video analysis tasks.

We explore a novel way to capture the local cross-temporal interaction. Same as the correlation operation, when calculating the visual similarity, we convert the global operation to a local operation to improve the efficiency and accuracy. This means the weight of the temporal aggregated feature F_i is calculated by only considering the temporal interaction with its k neighbors as Eq. 5:

$$w_i = \sum_{j=1}^{k} (w_i^j F_i^j); F_i^j \in \Omega_i^k; \tag{5}$$

where Ω_i^k indicates the set of k adjacent temporal features of F_i. Accordingly, a band matrix W_k is employed to learn the temporal attention, where W_k is computed as:

$$W_k = \begin{bmatrix} w^{1,1} & w^{1,k} & 0 & 0 & 0 \\ 0 & w^{2,2} & w^{2,k+1} & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & w^{T,T-k+1} & w^{T,T} \end{bmatrix} \tag{6}$$

Notably, W_k involves only $k \times T$ parameters, which are less than $T \times T$. Besides, we make all temporal channels to share the same learning parameters to boost the calculating speed:

$$w_i = \sum_{j=1}^{k} (w^j F_i^j); F_i^j \in \Omega_i^k \tag{7}$$

The range of the temporal interaction (the size of k) needs to be determined carefully. Following [50], the value of k can be decided by:

$$k = \psi(T) = \left| \frac{1}{\gamma} \log_2(T) + \frac{b}{\gamma} \right|_{odd}; \tag{8}$$

where $|v|_{odd}$ indicates the nearest odd number of v. We set both γ and b to 1 in our experiments.

Since the current frame and its adjacent frames have both the longest and shortest scales at the same time, the current temporal receptive field is further expanded. As a result, the features have learned the information of both the fast-tempo and the slow-tempo. Taking the interaction of the temporal dynamics at different temporal scales into account, our temporal attention module can better enhance the useful slow and fast visual tempo information and suppress the unnecessary ones.

C. Implementation

In MTDM, we first apply a 1×1 convolution to reduce channels to boost the computational efficiency. The C++/Cuda implemented version of the correlation operation in FlowNet [25] is adopted for our correlation tensor calculation. The motion estimation method of [20] is introduced to estimate the displacement map from the correlation tensor. In TAM, six $1 \times 3 \times 3$ depth-wise separable convolutions are used to exploit fine-grained multi-scale temporal semantics. For the temporal attention, it can be performed by a fast 1D convolution with a kernel size k, and then we extend the channel attention method ECA [50] to the temporal dimension. We utilize ResNet50 [33] as the backbone, whose structure is presented in Table I.

TABLE I
THE ILLUSTRATION OF OUR USED 2D BACKBONE RESNET50. NOTE THAT BOTH THE KERNEL SIZE AND THE OUTPUT SIZE ARE IN $W \times H$.

Stage	Layer	Output size	
raw		224	224
$conv_1$	$7 \times 7, 64$, stride 2,2	112	112
$pool_1$	3×3 max, stride 2,2	56	56
res_2	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	56	56
res_3	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	28	28
res_4	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$	14	14
res_5	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	7	7
global average pool, fc		1	1

IV. EXPERIMENTS

We evaluate the proposed method on various action recognition datasets, including Kinetics-400 [26], HMDB-51 [27], UCF-101 [28], and Something-Something V1 & V2 [29]. The baseline method in our experiments is TSM [32] which uses ResNet50 [33] without non-local modules [15], thus it is fair for comparison. Furthermore, we test our method

on multiple action recognition backbone networks (i.e. TSM, TEA, and I3D), and conduct plenty of ablation studies about the components of TCM on Something-Something V1, to analyze the effectiveness of the proposed TCM and its two components (i.e. MTDM and TAM). It should be noted that we focus on the gain of action recognition from the extraction of action visual tempo patterns, and we only use RGB frames rather than optical flow to save the computation cost.

Datasets. As mentioned in the previous work [30], the primary public datasets for action recognition can be roughly classified into two categories: (1) the temporal-related datasets (e.g. Something-Something V1 & V2 [29], in which the temporal motion interaction of objects should be emphasized for better action understanding. (2) The scene-related datasets (e.g. Kinetics-400 [26], UCF-101 [28] and HMDB-51 [27]), in which the temporal relation is less important compared to the temporal-related datasets, this is because the background information contributes more for determining the action label in most of the videos. The Something-Something V1 & V2 datasets focus on human interactions with daily life objects, thus classifying these interactions required to pay more attention to the temporal information. Consequently, the proposed method is mainly evaluated on Something-Something V1 & V2 as our goal is to improve the temporal modeling ability. Additionally, we also report experimental results on the scene-related datasets Kinetics-400 [26], HMDB-51 [27], and UCF-101 [28]. Kinetics-400 contains 400 human action categories, and provides 240k training videos and 20k validation videos. In our experiments, due to some videos in Kinetics-400 are unavailable, we collected 238,798 videos for training and 19,852 videos for validation.

Training. In general, we adopted the training strategy same as TSM [32]. Our model is initialized with the ImageNet pre-trained weights of ResNet50 (see Table I). The training settings for the Kinetics-400, UCF-101, and HMDB-51 datasets are also the same as TSM [32]. For the Something-Something V1 & V2 datasets, the training parameters are: the epochs are 50, the batch size is 32, the initial learning rate is 0.01 (decays by 0.1 at epoch 30, 40 and 45), the weight decay is 5e-4, and the dropout is 0.5. At training, for each video, we sample a clip with 8 or 16 frames, resize them to the scale of 240×320, and then crop a 224×224 patch from the resized images. The scale jittering is used for data augmentation. The final prediction follows the standard protocol of TSN [41].

Evaluation. For the Something-Something V1 & V2 datasets, two kinds of testing schemes are used: 1) single-clip and center-crop, where only a center crop of 224×224 from a single clip is utilized for evaluation; 2) 10-clip and 3-crop, where three crops of 224×224 and 10 randomly-sampled clips are employed for testing. The first testing scheme is with high efficiency while the second one is for improving the accuracy with a denser prediction strategy. We evaluate both the single-clip prediction and the average prediction of 10 randomly-sampled clips. For the Kinetics-400 dataset, we evaluate the average prediction of uniformly-sampled 10 clips from each video. For the UCF-101 and HMDB-51 datasets, 2 uniformly-sampled clips from each video are selected for evaluation.

TABLE II
COMPARISON OF OUR METHOD "TSM+TCM" WITH TSM ON DIFFERENT DATASETS. SPECIFICALLY, 8 FRAMES ARE INPUT FOR TRAINING AT TESTING, 10 VIDEO-CLIPS FOR KINETICS-400, 2 VIDEO-CLIPS FOR HMDB-51 AND UCF-101, AND A SINGLE VIDEO CLIP FOR SOMETHING-SOMETHING V1 & V2.

Dataset	Model	Top-1(%)	Top-5(%)	Top-1(%)
Kinetics-400	TSM	74.1	91.2	+1.5
	Ours	75.6	92.5	
UCF-101	TSM	95.9	99.7	+1.3
	Ours	97.2	99.8	
HMDB-51	TSM	73.5	94.3	+4.1
	Ours	77.6	96.4	
Sth-Sth V1	TSM	47.3	76.2	+4.7
	Ours	52.0	80.4	
Sth-Sth V2	TSM	61.7	87.4	+1.7
	Ours	63.4	88.6	

A. Performance on CNN baselines

TCM can be seamlessly injected into a CNN baseline to significantly enhance its temporal information modeling ability. To demonstrate that the enhancement is generalized and steady, we compare our TCM with some baseline networks on some famous action recognition benchmarks.

1) **Evaluation on Different Datasets.** In this experiment, as analyzed before, we select the representative model TSM [32] as the CNN baseline, and use the same training and testing protocols for both the original TSM [32] and the modified model "TSM+TCM" for fair comparison. The results are shown in Table II. In the upper part, for the datasets Kinetics-400, UCF-101, and HMDB-51, their temporal information is relatively less important. In contrast, in the lower part, for the datasets Something-Something V1 & V2, the temporal information becomes very important. When integrating our TCM module into TSM, the performance of "TSM+TCM" has significantly improved on both the scene dominant datasets and the temporal dominant datasets. For example, compared with TSM, on the large-scale dataset Kinetics-400, the Top-1 accuracy of "TSM+TCM" is improved by 1.5% (75.6% vs. 74.1%); On the relatively smaller-scale datasets UCF-101 and HMDB-51, the Top-1 accuracy of "TSM+TCM" is boosted respectively by 1.3% and 4.1%; On the temporal dominated datasets Something-Something V1 & V2, the performance improvement is more obvious, where the Top-1 accuracy is enhanced separately by 4.7% and 1.7%. This proves that the proposed TCM is effective to improve the temporal modeling ability of the baseline.

2) **Evaluation on Different Backbones.** We apply our TCM to a variety of backbone networks, and show their accuracy on the Something-Something V1 dataset in Table III. There are two parts: the upper part is 2D-CNN methods, and the lower part is 3D-CNN methods. Particularly, in all networks, TCM is placed right behind layer$_{res3}$. For TSM over different backbones, "TSM+TCM" can clearly enhance the accuracy of action recognition with only a small increase in parameters. Compared to the baseline TSM-ResNet50, TCM obtains a significant gain of about 6.4% (52.0% vs. 45.6%) at Top-1 accuracy at the cost of only 5.6% (35.3G vs. 33.4G) and 0.8% (24.5M vs. 24.3M) growth in FLOPs and parameters. For the well-performed TEA [31] which can stimulate and aggregate

TABLE III
COMPARISON BETWEEN OUR TCM AND OTHER BACKBONES ON THE SOMETHING-SOMETHING V1 DATASET.

	Method	TCM	Input			FLOPs	Params	Sth-Sth V1	
								Top-1(%)	Top-5(%)
2D	TSM-ResNet18 [41]		224	224	8	14.5G	11.3M	42.8	72.3
		X	224	224	8	16.4G$_{(+1:9)}$	11.5M$_{(+0:2)}$	45.8$_{(+3:0)}$	74.8$_{(+2:5)}$
	TSM-ResNet50 [41]		224	224	8	33.4G	24.3M	45.6	74.2
		X	224	224	8	35.3G$_{(+1:9)}$	24.5M$_{(+0:2)}$	52.0$_{(+6:4)}$	80.4$_{(+6:2)}$
	TSM-ResNet101 [41]		224	224	8	63.1G	42.9M	46.3	75.8
		X	224	224	8	65.0G$_{(+1:9)}$	43.1M$_{(+0:2)}$	52.6$_{(+6:3)}$	81.4$_{(+5:6)}$
	TEA [42]		224	224	8	34.7G	24.4M	48.9	78.1
		X	224	224	8	36.6G$_{(+1:9)}$	24.6M$_{(+0:2)}$	50.6$_{(+1:7)}$	79.7$_{(+1:6)}$
3D	3D-ResNet-18 [37]		112	112	16	33.2G	33.3M	16.4	45.3
		X	112	112	16	33.4G$_{(+0:2)}$	33.5M$_{(+0:2)}$	19.7$_{(+3:3)}$	48.5$_{(+3:2)}$
	3D-ResNet-50 [37]		112	112	16	40.4G	46.2M	22.3	51.4
		X	112	112	16	40.6G$_{(+0:2)}$	46.4M$_{(+0:2)}$	24.1$_{(+1:8)}$	53.7$_{(+2:3)}$
	3D-ResNeXt from [51]		112	112	16	9.6G	47.8M	24.5	53.2
		X	112	112	16	9.8G$_{(+0:2)}$	48.2M$_{(+0:2)}$	26.1$_{(+1:6)}$	55.7$_{(+2:5)}$

the temporal information effectively, our TCM also boosts its performance for a large margin, e.g. the Top-1 accuracy enhances from 48.9% to 50.6%. For 3D-ResNet [37] over different depth, TCM can steadily promote the performance, e.g. 3D-ResNet-18 (Top-1 + 3.3%), 3D-ResNet-50 (Top-1 + 1.8%). For the famous 3D network 3D-ResNeXt [51], our TCM further improves its temporal modeling capability, where the Top-1 accuracy is modified by 1.6% (24.5% vs. 26.1%).

B. Comparison with state-of-the-arts

To evaluate the temporal modeling ability and the entire capacity of our method, we compare our TCM-derived models with the state-of-the-arts extensively on both the temporal-dominated datasets Something-Something V1 & V2 and the scene dominated datasets Kinetics-400, UCF-101 and HMDB-51. The results are reported in Table IV, Table V and Table VI.

Table IV shows the performance of 23 recent action recognition methods on the Something-Something V1 & V2 datasets. There are three parts in this table: 3D CNN methods [52], [53], [54], [55] (in the upper part), 2D CNN methods [41], [42], [60], [61], [30], [31], [20] (in the middle part), and the proposed TCM based methods (in the bottom part). Without any bells and whistles, the Top-1 accuracy of our "TSM+TCM" method i.e. "TCM-R50" on the Something-Something V1 dataset reaches to 52.2%, which surpasses its 2D CNN based counterparts STM [30], TEA [31], and TANet [62] that need double input (16 input frames) for at least 0.3%. Importantly, with 16 input frames, the accuracy of our "TSM+TCM" method on the Something-Something V1 & V2 datasets is further improved, where its Top-1 accuracy is higher than STM [30], TEA [31], and TANet [62] for at least 1.2% on Something-Something V1 and 0.9% on Something-Something V2. In addition, compared to the 3D CNN based methods,e.g. 3D DenseNet121 [57], the Top-1 accuracy is improved by 2.9% (53.1% vs. 50.2%) and 2.2% (65.1% vs. 62.9%) separately on the Something-Something V1 dataset and the Something-Something V2 dataset. Compared to NL I3D [53], the performance is improved by 7.8% with using less input (8 vs. 32) and less computation (35GFlops vs. 168GFlops 2).

Following [32], [20], we ensemble our 8-frame and 16-frame models by averaging their prediction scores. Our 10-clip model obtains remarkable results on the Something-Something V1 dataset. Specically, as shown in the last row of Table IV, its Top-1 and Top-5 accuracy outperforms the state-of-the-art method TDN [65] by 0.4% (57.2% vs. 56.8%) and 1.1% (85.2% vs. 84.1%), respectively. Furthermore, on the Something-Something V2 dataset, in contrast to TDN [65], our 10-clip model also has a better performance in the Top-5 accuracy (92.2% vs. 91.6%) and its Top-1 accuracy is just a little bit lower (67.8% vs. 68.2%).

Table V shows the comparison with the state-of-the-art approaches on the scene dominated dataset Kinetics-400. It can be clearly seen that our TCM has an outstanding performance. Firstly, our 8-frame TCM-R50 surpasses the 64-frame I3D method [39] (Top-1 accuracy: 76.1% vs. 72.1%), and it achieves a competitive accuracy to the 128-frame Nonlocal-R50 approach [15] (Top-1 accuracy: 76.1% vs. 76.5%) while its GFlops is 8 less. Moreover, our 8-frame TCM-R50 model performs even better than the 8-frame SlowOnly method [8] (Top-1 accuracy: 76.1% vs. 74.8%) with 1.2less GFlops. All these results demonstrate that, our TCM network, is more accurate and efficient than the non-local network to model the temporal relationships for video classification. Secondly, our 16-frame TCM-R50 outperforms most of its 16-frame counterparts,i.e. STM [30] (Top-1 accuracy: 77.4% vs. 73.7%), TEA [31] (Top-1 accuracy: 77.4% vs. 76.1%), MSNet [20] (Top-1 accuracy:77.4% vs. 76.4%), and TEINet [63] (Top-1 accuracy: 77.4% vs. 76.2%). With 5.1less GFlops than the 128-frame Nonlocal-R101 method [15], our model obtains a comparable accuracy (Top1 acc: 77.4% vs. 77.7%). Thirdly, we perform the score fusion over 8-frame TCM-R50 and 16-frame TCM-R50, which mimics the two-steam fusion with two temporal rates. At testing, we use 10 clips and 3 crops per clip. Our TCM achieves a higher accuracy than the "8+32-frame" SlowFast model [8], with using less input frames and a little bit less GFlops (105 vs. 106). Table V reveals that, spatio-temporal learning of our TCM is more effective than temporal shift of TSM.

To further verify the generalization ability of the explored TCM, we transfer the trained 16-frame TCM-R50 model

TABLE IV
PERFORMANCE COMPARISON WITH STATE-OF-THE-ARTS ON THE SOMETHING-SOMETHING V1 & V2 DATASETS. MOST OF THE RESULTS ARE COPIED FROM THE CORRESPONDING PAPER, AND THE SYMBOL "-" DENOTES THE RESULT IS NOT GIVEN.

Method	Frame	FLOPs	clips	Params	Sth-Sth V1		Sth-Sth V2	
					Top-1(%)	Top-5(%)	Top-1(%)	Top-5(%)
ECO_{En} Lite [52]	92	267	1	150M	46.4	-	-	-
I3D from [53]	32	153G	2	28.0M	41.6	72.2	-	-
NL-I3D from [53]	32	168G	2	35.3M	44.4	76.0	-	-
NL-I3D + GCN [53]	32	303G	2	62.2M	46.1	76.8	-	-
S3D-G [54]	64	71G	1	11.6M	48.2	78.7	-	-
DFB-Net [55]	16	N/A	1	-	50.1	79.5	-	-
CorrNet-101 [56]	32	224G	30	-	51.7	-	-	-
3D DenseNet121 [57]	16	31G	1	21.4M	50.2	78.9	62.9	88.0
CIDC(I3D) [58]	32	92G	30	87M	-	-	56.3	83.7
RubiksNet [59]	8	33G	1	-	46.4	74.5	58.8	85.6
TSN [41]	8	16G	1	10.7M	19.5	-	33.4	-
TRN [42]	8	16G	N/A	18.3M	34.4	-	48.8	-
MFNet [60]	10	N/A	10	-	43.9	73.1	-	-
CPNet [61]	24	N/A	96	-	-	-	57.7	84.0
STM [30]	16	67G	30	24.0M	50.7	80.4	64.2	89.8
TSM [32]	16+8	98G	1	48.6M	49.7	78.5	62.9	88.1
TEA [31]	16	70G	1	24.5M	51.9	80.3	-	-
TANet [62]	16+8	99G	1	25.6M	50.6	79.3	-	-
MSNet [20]	16+8	101G	10	49.2M	55.1	84.0	67.1	91.0
CIDC(R2D) [58]	32	72G	30	85M	-	-	40.2	68.6
TEINet [63]	16+8	99G	1	30.4M	52.5	-	65.5	89.8
ACTION-Net [64]	16	69.5G	1	28.1M	-	-	64.0	89.3
TDN [65]	16+8	198G	1	-	56:8	84.1	68:2	91.6
TCM-R50	8	35G	1	24.5M	52.2	80.4	63.5	88.7
TCM-R50	16	70G	1	24.5M	53.1	81.2	65.1	89.6
$TCM\text{-}R50_{En}$	16+8	105G	1	49.0M	54.7	82.6	66.7	90.7
$TCM\text{-}R50_{En}$	16+8	105G	10	49.0M	57:2	85:2	67:8	92:2

TABLE V
PERFORMANCE COMPARISON WITH THE STATE-OF-THE-ARTS ON THE KINETICS-400 DATASET. THE SYMBOL "N/A" DENOTES THE RESULT THAT IS NOT GIVEN.

Method	Pretrain	Frame	FLOPs	Views	Kinetics-400	
					Top-1(%)	Top-5(%)
ECO [52]	Scratch	92	N/A	N/A	70.0	89.4
I3D [39]	Scratch	64	108G	N/A	72.1	90.3
Two-Stream I3D [39]	Scratch	64	216G	N/A	75.7	92.0
R(2+1)D [40]	Sports-1M	32	152G	10	74.3	91.4
ARTNet [66]	Scratch	16	23.5G	250	69.2	88.3
S3D-G [54]	ImageNet	64	66.4G	N/A	77.2	93.0
Nonlocal-R50 [15]	ImageNet	128	282G	30	76.5	92.6
Nonlocal-R101 [15]	ImageNet	128	359G	30	77.7	93.3
ip-CSN [67]	ImageNet	32	83G	30	76.7	92.3
CIDC(I3D) [58]	ImageNet	32	92G	30	74.5	91.3
TPN [6]	Scratch	32	N/A		78.9	93.9
SmallBigNet [68]	Scratch	32	418G	12	77.4	93.3
CorrNet [56]	Scratch	32	224G	30	79.2	N/A
SlowOnly [8]	Scratch	8	41.9G	30	74.8	91.6
SlowFast [8]	Scratch	8+32	106G	30	77.9	93.2
SlowFast [8]	Scratch	16+64	234G	30	79:8	93.9
X3D [4]	Scratch	16	48.4G	30	79.1	93.9
TSN [41]	ImageNet	25	3.2G	10	72.5	90.5
TSM [32]	ImageNet	16	65G	30	74.7	91.4
STM [30]	ImageNet	16	67G	30	73.7	91.6
TEA [31]	ImageNet	16	70G	30	76.1	92.5
MSNet [20]	ImageNet	16	67G	30	76.4	N/A
CIDC(R2D) [58]	ImageNet	32	72G	30	72.2	90.1
TEINet [63]	ImageNet	16	66G	30	76.2	92.5
TANet-152 [62]	ImageNet	16	242G	12	79.3	94.1
TDN [65]	ImageNet	16+8	198G	30	79.4	94:4
TCM-R50	ImageNet	8	35G	30	76.1	92.3
TCM-R50	ImageNet	16	70G	30	77.4	93.1
$TCM\text{-}R50_{En}$	ImageNet	16+8	105G	30	78:5	93:8

TABLE VI
COMPARISON WITH THE STATE-OF-THE-ARTS ON THE UCF-101 AND HMDB-51 DATASETS. THE SYMBOL "-" DENOTES THE RESULT THAT IS NOT GIVEN.

Method	Pretrain	Backbone	UCF-101	HMDB-51
TSN [41]	ImageNet	Inception V2	86.4%	53.7%
P3D [69]	ImageNet	ResNet50	88.6%	
C3D [36]	Sports-1M	ResNet18	85.8%	54.9%
I3D [39]	Kinetics	Inception V2	95.6%	74.8%
ARTNet [66]	Kinetics	ResNet18	94.3%	70.9%
S3D [54]	Kinetics	Inception V2	96.8%	75.9%
R(2+1)D [40]	Kinetics	ResNet34	96.8%	74.5%
TSM [32]	Kinetics	ResNet50	96.0%	73.2%
STM [30]	Kinetics	ResNet50	96.2%	72.2%
TEA [31]	Kinetics	ResNet50	96.9%	73.3%
TDN [65]	Kinetics	ResNet50	97.4%	76.3%
TCM-R50 (ours)	Kinetics	ResNet50	97.1%	77.5%

TABLE VII
PERFORMANCE COMPARISON WITH TPN ON THE SOMETHING-SOMETHING V1 & V2 DATASETS. SPECIFICALLY, 8 FRAMES ARE INPUT FOR TRAINING. SINGLE-CLIP AND CENTER-CROP TESTING SCHEME IS USED HERE

Method	FLOPs	Params	Top1@V1	Top1@V2
TSM-R50 [32]	33.4G	24.3M	45.6	59.1
TSM+TPN [6]	41.5G	82.5M	49.0	62.0
TSM+TCM(ours)	35.3G	24.5M	52.2	63.5

from the Kinetics-400 dataset to the UCF-101 and HMDB-51 datasets same as the previous works [32], [31], [65]. We follow the standard evaluation metric on the two datasets and report the average Top-1 accuracy over the three splits, where the results are summarized in Table VI. We compare our TCM with the advanced methods such as the 2D baseline TSM [32], the 3D CNN based methods I3D [39], C3D [36], and R(2+1)D [40], and the other temporal modeling methods [30], [31], [65]. From the results, we can see that our TCM is superior to these methods, and the performance improvement is more obvious on the HMDB51 dataset which is boosted by at leat 1.2% (Top-1 accuracy: 77.5% vs. 76.3%). The human actions in HMDB51 are more relevant with motion information, therefore temporal modeling is more important on this dataset. On the other side, on the UCF-101 dataset, our TCM-R50 also achieves competitive result to the first place (Top-1 accuracy: 97.1% vs. 97.4%).

C. Comparison with TPN

To make a fair comparison, we compare the performance of our TCM and TPN [6] on the Something-Something V1 & V2 datasets, using the same backbone (TSM-R50 [32]) and same experiment settings. Results in Table VII demonstrate that TCM is superior to TPN: 1) Higher accuracy, where it outperforms TPN on the Something-Something V1 dataset by 3.2% and the Something-Something V2 dataset by 1.5%; 2) Fewer parameters, where its parameters is less than 30% of TPN (TCM vs. TPN: 24.5M vs. 82.5M). 3) Less computation, where it takes only about 85% Flops of TPN (TCM vs. TPN: 35.3G vs. 41.5G). The efficiency of TPN is not satisfactory may due to the use of 3D convolution. TCM is more efficient since it is based on 2D convolution.

D. Ablation Study

Ablation studies about the components of our TCM are conducted on the Something-Something V1 dataset. Particularly, the ResNet-18 with the temporal shift module [32] is served as the backbone here. Following the setting of [32], 8 input frames, which are sampled from the video via the segment-based sampling method [41], are utilized for training and inference. The training parameters are: the training epochs are 40, the batch size is 64, the initial learning rate is 0.02 (decays by 0.1 at epoch 20 & 30), the weight decay is 5e-4, and the dropout is 0.5.

1) Which feature source is most suitable for building a multi-scale temporal motion pyramid?: Extensive experiments on feature sources are tested, results shown in Table VIII verify that the proposed TCM overcomes the previous approach's inability to explore benefits from relatively shallow sources, e.g. res2 or res3. Even for high-level feature sources, significant improvement is gained. Enjoying the flexibility to plug-and-play in a single-layer, the multi-layer style can also be performed by using multiple TCM modules simultaneously. Its performance is slightly improved compared to the baseline but is degraded in contrast to the usage of a single TCM. This is because stacking multiple TCM modules damages the brightness consistency of the prior TCM layer. Since res2 is too shallow to extract enough spatial features, the accuracy increases the most when TCM is placed right after res3 layer. As a result, we select to use a single TCM right behind layer res3 finally.

TABLE VIII
PERFORMANCE COMPARISON WITH DIFFERENT FEATURE SOURCES. TCM IS PLACED RIGHT BEHIND THE SPECIFIED LAYER

Layer	GFLOPs	Top-1(%)	Top-5(%)
baseline	14.5	42.8	72.3
res2	17.5	42.9	72.6
res3	16.4	45.9	75.7
res4	15.1	43.8	73.1
res5	14.7	43.6	72.8
res{2,3}	19.4	45.8	75.6
res{2,3,4}	20.0	45.8	75.5
res{2,3,4,5}	20.2	45.6	75.4

2) How important of MTDM and TAM?: MTDM is used for multi-scale temporal dynamics extraction and TAM is used for temporal scales aggregation. The effect of these modules is studied in Table IX. From which we can observe: both MTDM and TAM can enhance the accuracy of action recognition largely, and the action recognition performance is further boosted when combining MTDM with TAM.

TABLE IX
PERFORMANCE COMPARISON WITH DIFFERENT TCM COMPONENTS

MTDM	TAM	Top-1(%)	Top-1(%)
		42.6	baseline
X		43.8	+1.2
	X	43.2	+0.5
X	X	45.9	+3.3

3) What is the difference with the existing motion cue learning method?: As far as we known, the existing motion

cue learning methods [70], [20], [19], [30], [56] mainly aimed at extracting the feature-level motion patterns between adjacent frames. TVNet [70] and Rep-Flow [70], [19] internalized the TV-L1 optical flow in their networks, which enabling to capture the motion information and appearance information in an end-to-end way. STM [30], TEA [31], and CorrNet [56] propose approaches to establish frame-to-frame matches over convolutional feature maps, and well-designed blocks for learning better temporal information are applied to replace the original residual blocks in the ResNet architecture to construct a simple yet effective network. MSNet [20] presents a trainable module named MotionSqueeze to substitute the external and heavy computational optical flow with the internal and lightweight learned motion features. These works have promoted video understanding, but the extraction of action visual tempo from video sequence features has been rarely accounted. Our work aims to fill this gap, where the video sequence features are exploited to capture the movement patterns on different temporal scales, and the useful visual tempo information is enhanced adaptively. We compared our method with these motion cue learning methods on the Kinetics-400 dataset in Table X, it reveals that our method is superior to these methods due to it concerns multi-scale fine-grained temporal dynamics of both the fast-tempo and the slow-tempo.

TABLE X
PERFORMANCE COMPARISON WITH THE EXISTING MOTION CUE LEARNING METHODS ON THE KINETICS-400 DATASET. THE SYMBOL "-" DENOTES THE RESULT THAT IS NOT GIVEN

Method	Frame	GFLOPs	Clips	FPS	Top-1(%)
TVNet [70]	18	N/A	250	-	68.5[1]
Rep-flow [19]	32	152[2]	25	2.0	75.5
STM [30]	16	67	30	-	73.7
TEA [31]	16	70	1	-	74.0
CorrNet-50 [56]	32	115	10	-	77.2
MSNet [20]	16	67	10	31.2	76.4
TCM(ours)	16	70	10	28.1	77.2

[1] represents that the results of the method are reproduced by us according to the source code they supplied.

TABLE XI
PERFORMANCE COMPARISON WITH THE PLUG-IN-AND-PLAY MODULES ON THE SOMETHING-SOMETHING V2 DATASET (TSM-R50 IS USED AS A BASELINE). ALL METHODS USE RESNET50 PRE-TRAINED ON IMAGENET AS THE BACKBONE AND 8-FRAME INPUT FOR FAIR COMPARISON. THE LEAST PARAM. AND THE BEST RESULT ARE HIGHLIGHTED AS BOLD THE SYMBOL "-" DENOTES THE RESULT THAT IS NOT GIVEN

Method	GFLOPs	Param.	Sth-Sth V2	
			Top-1(%)	Top-5(%)
baseline(TSM-R50 [32])	32.8	24.3M	58.8	85.4
Nonlocal TSM-R50 [32]	49.3	31.2M	60.1	86.7
TEINet [63]	33	30.4M	61.3	-
MSNet [20]	34.3	24.6M	63.0	88.4
ACTION-Net [64]	34.7	28.1M	62.5	87.3
TCM-R50(ours)	35.3	24.5M	**63.5**	**88.7**

4) Compared with plug-in-and-play modules: We make a comprehensive comparison with methods [32], [63], [20], [64], which enjoy a plug-and-play manner likes our TCM, on the Something-Something V2 dataset, the results are shown in Table XI. The TSM-R50 [32] method is served as the baseline here for performance and efficiency comparison. For the Nonlocal TSM-R50 method [32], we retrained the model on the Something-Something V2 dataset via the official PyTorch code base [32]. Compared to Nonlocal TSM-R50, our TCM-R50 uses only 0.7 GFLops and 0.8 vParam., and achieves much better performance (Top1 accuracy: 63.5% vs. 60.1%, Top5 accuracy: 88.7% vs. 85.4%). Probably because the non-local module focuses on capturing global dependencies, in contrast, our approach pays close attention to extract the most appropriate dependencies. In the spatial dimension, pixel-wise optical-flow-like motion information is obtained by an optimized MotionSqueeze [20] algorithm. In the temporal dimension, for the first time, we sample the learned features with different ratios, and carry out an efficient temporal attention consideration that involves cross-temporal interaction. Compared to other well-designed approaches, our TCM-R50 utilizes the fewest parameters and gets the highest performance improvement.

G. Visualization

To further explore the working mechanism of our TCM, we visualize the class activation maps with Grad-CAM [71], [72], [73]. Fig. 4 shows the feature visualizations that are characterized by TSN [41], TSM [32] and our TCM for the action "Moving something and something so they pass each other" on the Something-Something V1 dataset. In our visualization, we take 8 frames from the moment T0 to the moment T7 as input, and plot the activation maps for each frame. From the results, it can be found that 1) TSN only pays attention to objects and fails to capture the movement of objects and human hands; 2) TSM can capture the coarse motion information, but it is unable to accurately locate the action-relevant regions; 3) Our TCM is superior to the TSM baseline in focusing on the action-relevant regions, due to the long-term and short-term temporal modeling capacity of the proposed TCM.

F. Empirical Analysis

To study the robustness of TCM to action visual tempo variation, we follow [6] to evaluate the accuracy drop by re-sampling the input frames with different temporal intervals. We first train TSM-ResNet18 and TCM-ResNet18 on the Something-Something V1 dataset with 4 (frames interval) inputs, then we re-scale the original 4 input by re-sampling the frames with the stride equals to of 2,3,4,5,6,7,8 respectively, accordingly the temporal scales of a given action instance are adjusted. For some videos with insufficient number of frames, we copy the last frame until the number of the input frames is reached. Fig. 5 shows the accuracy curves of varying the action temporal scales for TSM-ResNet18 and our TCM-ResNet18. Clearly, our TCM improves the robustness of the baseline, and gets a smoother curve (see the orange curve on the top), which strongly supports the conclusion that our TCM can effectively extract and fuse the action visual tempo features.

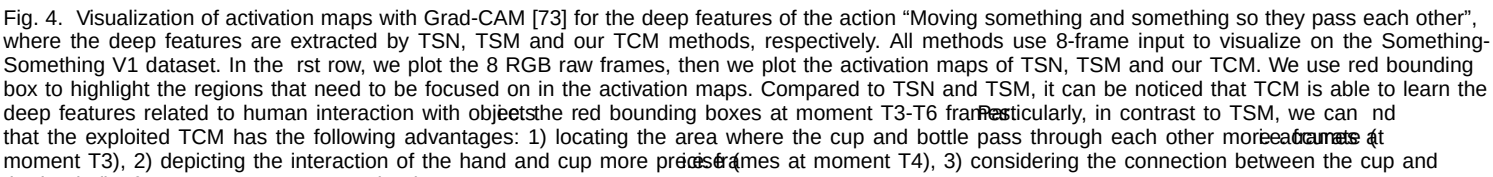


Fig. 4. Visualization of activation maps with Grad-CAM [73] for the deep features of the action “Moving something and something so they pass each other”, where the deep features are extracted by TSN, TSM and our TCM methods, respectively. All methods use 8-frame input to visualize on the Something-Something V1 dataset. In the first row, we plot the 8 RGB raw frames, then we plot the activation maps of TSN, TSM and our TCM. We use red bounding box to highlight the regions that need to be focused on in the activation maps. Compared to TSN and TSM, it can be noticed that TCM is able to learn the deep features related to human interaction with objects. Specifically, in contrast to TSM, we can find accurate (i.e. the red bounding boxes at moment T3-T6 frames) that the exploited TCM has the following advantages: 1) locating the area where the cup and bottle pass through each other more accurately (i.e. frames at moment T3), 2) depicting the interaction of the hand and cup more precisely (i.e. frames at moment T4), 3) considering the connection between the cup and the bottle (i.e. frames at moment T5 and T6).

V. Conclusion

We propose a novel Temporal Correlation Module (TCM) to deal with the variation of action visual tempo in videos, which includes a Multi-scale Temporal Dynamics Module (MTDM) and a Temporal Attention Module (TAM). MTDM extracts pixel-wise fine-grained temporal dynamics for both the fast-tempo and the slow-tempo by utilizing a correlation operation. TAM adaptively selects and enhances the most effective action visual tempo information by taking across-temporal dynamics interaction into account. The explored TCM can be seamless integrated into the current action recognition backbones and optimized in an end-to-end way to capture the action visual tempo commendably. It is especially effective when incorporating it to the low-level layer of the backbone. Extensive experiments on 5 representative datasets have demonstrated the effectiveness of TCM in both accuracy and efficiency.

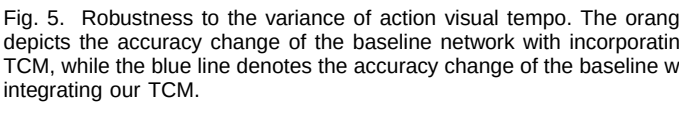


Fig. 5. Robustness to the variance of action visual tempo. The orange line depicts the accuracy change of the baseline network with incorporating our TCM, while the blue line denotes the accuracy change of the baseline without integrating our TCM.

Acknowledgment

This work was supported by the National Natural Science Foundation of China under Grant 62106177. It was also supported by the Central University Basic Research Fund of China (No.2042020KF0016). The numerical calculation was supported by the supercomputing system in the Supercomputing Center of Wuhan University.

REFERENCES

[1] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Proc. of International Conference on Neural Information Processing Systems*, 2014, pp. 568–576.

[2] Y. Du, Y. Fu, and L. Wang, "Representation learning of temporal dynamics for skeleton-based action recognition," *IEEE Transactions on Image Processing*, vol. 25, no. 7, pp. 3010–3022, 2016.

[3] R. Girdhar, J. Carreira, C. Doersch, and A. Zisserman, "Video action transformer network," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 244–253.

[4] C. Feichtenhofer, "X3d: Expanding architectures for efficient video recognition," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 203–213.

[5] A. Arnab, M. Dehghani, G. Heigold, C. Sun, M. Lučić, and C. Schmid, "Vivit: A video vision transformer," in *Proc. of IEEE International Conference on Computer Vision*, 2021, pp. 6836–6846.

[6] C. Yang, Y. Xu, J. Shi, B. Dai, and B. Zhou, "Temporal pyramid network for action recognition," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 591–600.

[7] D. Zhang, X. Dai, and Y.-F. Wang, "Dynamic temporal pyramid network: A closer look at multi-scale modeling for activity detection," *Asian Conference on Computer Vision*, 2018, pp. 712–728.

[8] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "Slowfast networks for video recognition," in *Proc. of IEEE International Conference on Computer Vision*, 2019, pp. 6202–6211.

[9] C. Yang, Y. Xu, B. Dai, and B. Zhou, "Video representation learning with visual tempo consistency," *arXiv preprint arXiv:2006.15489*, 2020.

[10] F. S. Khan, J. van de Weijer, R. M. Anwer, M. Felsberg, and C. Gatta, "Semantic pyramids for gender and action recognition," *IEEE Transactions on Image Processing*, vol. 23, no. 8, pp. 3633–3645, 2014.

[11] H. Cholakkal, J. Johnson, and D. Rajan, "Backtracking spatial pyramid pooling-based image classifier for weakly supervised top–down salient object detection," *IEEE Transactions on Image Processing*, vol. 27, no. 12, pp. 6064–6078, 2018.

[12] G. Chen, T. Gu, J. Lu, J.-A. Bao, and J. Zhou, "Person re-identification via attention pyramid," *IEEE Transactions on Image Processing*, vol. 30, pp. 7663–7676, 2021.

[13] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[14] W. Luo, Y. Li, R. Urtasun, and R. Zemel, "Understanding the effective receptive field in deep convolutional neural networks," *Advances in neural information processing systems*, vol. 29, 2016.

[15] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[16] L. Tian, Z. Tu, D. Zhang, J. Liu, B. Li, and J. Yuan, "Unsupervised learning of optical flow with cnn-based non-local filtering," *IEEE Transactions on Image Processing*, vol. 29, pp. 8429–8442, 2020.

[17] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume," in *Proc. of IEEE conference on computer vision and pattern recognition*, 2018, pp. 8934–8943.

[18] T.-W. Hui, X. Tang, and C. C. Loy, "A lightweight optical flow cnn—revisiting data fidelity and regularization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 8, pp. 2555–2569, 2021.

[19] A. J. Piergiovanni and M. S. Ryoo, "Representation flow for action recognition," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9945–9953.

[20] H. Kwon, M. Kim, S. Kwak, and M. Cho, "Motionsqueeze: Neural motion feature learning for video understanding," *Proc. of European Conference on Computer Vision*, 2020, pp. 345–362.

[21] G. Chen, Y. Rao, J. Lu, and J. Zhou, "Temporal coherence or temporal motion: Which is more critical for video-based person re-identification?" in *Proc. of European Conference on Computer Vision*, 2020, pp. 660–676.

[22] Y.-D. Zheng, Z. Liu, T. Lu, and L. Wang, "Dynamic sampling networks for efficient action recognition in videos," *IEEE Transactions on Image Processing*, vol. 29, pp. 7970–7983, 2020.

[23] Y. Ji, Y. Zhan, Y. Yang, X. Xu, F. Shen, and H. T. Shen, "A context knowledge map guided coarse-to-fine action recognition," *IEEE Transactions on Image Processing*, vol. 29, pp. 2742–2752, 2019.

[24] Z. Liu, L. Wang, W. Wu, C. Qian, and T. Lu, "Tam: Temporal adaptive module for video recognition," in *Proc. of IEEE International Conference on Computer Vision*, 2021, pp. 13 708–13 718.

[25] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *Proc. of IEEE international conference on computer vision*, 2015, pp. 2758–2766.

[26] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev *et al.*, "The kinetics human action video dataset," *arXiv preprint arXiv:1705.06950*, 2017.

[27] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "HMDB: a large video database for human motion recognition," in *Proc. of International Conference on Computer Vision*, 2011.

[28] K. Soomro, A. R. Zamir, and M. Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," *arXiv preprint arXiv:1212.0402*, 2012.

[29] R. Goyal, S. Ebrahimi Kahou, V. Michalski, J. Materzynska, S. Westphal, H. Kim, V. Haenel, I. Fruend, P. Yianilos, M. Mueller-Freitag *et al.*, "The" something something" video database for learning and evaluating visual common sense," in *Proc. of IEEE International Conference on Computer Vision*, 2017, pp. 5842–5850.

[30] B. Jiang, M. Wang, W. Gan, W. Wu, and J. Yan, "Stm: Spatiotemporal and motion encoding for action recognition," in *Proc. of IEEE International Conference on Computer Vision*, 2019, pp. 2000–2009.

[31] Y. Li, B. Ji, X. Shi, J. Zhang, B. Kang, and L. Wang, "Tea: Temporal excitation and aggregation for action recognition," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 909–918.

[32] J. Lin, C. Gan, and S. Han, "Tsm: Temporal shift module for efficient video understanding," in *Proc. of IEEE International Conference on Computer Vision*, 2019, pp. 7083–7093.

[33] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. of IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[34] H. Yang, C. Yuan, L. Zhang, Y. Sun, W. Hu, and S. J. Maybank, "Sta-cnn: Convolutional spatial-temporal attention learning for action recognition," *IEEE Transactions on Image Processing*, vol. 29, pp. 5783–5793, 2020.

[35] Z. Tu, W. Xie, J. Dauwels, B. Li, and J. Yuan, "Semantic cues enhanced multimodality multistream cnn for action recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 5, pp. 1423–1437, 2018.

[36] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *Proc. of IEEE international conference on computer vision*, 2015, pp. 4489–4497.

[37] K. Hara, H. Kataoka, and Y. Satoh, "Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet?" in *Proc. of IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 6546–6555.

[38] M. Sabokrou, M. Fayyaz, M. Fathy, and R. Klette, "Deep-cascade: Cascading 3d deep neural networks for fast anomaly detection and localization in crowded scenes," *IEEE Transactions on Image Processing*, vol. 26, no. 4, pp. 1992–2004, 2017.

[39] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *Proc. of IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6299–6308.

[40] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, "A closer look at spatiotemporal convolutions for action recognition," in *Proc. of IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 6450–6459.

[41] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, "Temporal segment networks for action recognition in videos," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 11, pp. 2740–2755, 2019.

[42] B. Zhou, A. Andonian, A. Oliva, and A. Torralba, "Temporal relational reasoning in videos," in *Proc. of European Conference on Computer Vision*, 2018, pp. 803–818.

[43] S.-H. Gao, M.-M. Cheng, K. Zhao, X.-Y. Zhang, M.-H. Yang, and P. Torr, "Res2net: A new multi-scale backbone architecture," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 2, pp. 652–662, 2021.

[44] J. Liu, A. Shahroudy, M. Perez, G. Wang, L.-Y. Duan, and A. C. Kot, "Ntu rgb+ d 120: A large-scale benchmark for 3d human activity understanding," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 10, pp. 2684–2701, 2019.