STM: SpatioTemporal and Motion Encoding for Action Recognition

Boyuan Jiang *
Zhejiang University
byjiang@zju.edu.cn

MengMeng Wang †
SenseTime Group Limited
wangmengmeng@sensetime.com

Weihao Gan
SenseTime Group Limited
ganweihao@sensetime.com

Wei Wu
SenseTime Group Limited
wuwei@sensetime.com

Junjie Yan
SenseTime Group Limited
yanjunjie@sensetime.com

Abstract

Spatiotemporal and motion features are two complementary and crucial information for video action recognition. Recent state-of-the-art methods adopt a 3D CNN stream to learn spatiotemporal features and another flow stream to learn motion features. In this work, we aim to efficiently encode these two features in a unified 2D framework. To this end, we first propose an STM block, which contains a Channel-wise SpatioTemporal Module (CSTM) to present the spatiotemporal features and a Channel-wise Motion Module (CMM) to efficiently encode motion features. We then replace original residual blocks in the ResNet architecture with STM blcoks to form a simple yet effective STM network by introducing very limited extra computation cost. Extensive experiments demonstrate that the proposed STM network outperforms the state-of-the-art methods on both temporal-related datasets (i.e., Something-Something v1 & v2 and Jester) and scene-related datasets (i.e., Kinetics-400, UCF-101, and HMDB-51) with the help of encoding spatiotemporal and motion features together.

1. Introduction

Following the rapid development of the cloud and edge computing, we are used to engaged in social platforms and live under the cameras. In the meanwhile, various industries, such as in security and transportation, collect vast amount of videos which contain a wealth of information, ranging from people's behavior, traffic, and etc. Huge video information attracts more and more researchers to the video understanding field. The first step of the video understanding is action recognition which aims to recognize the human actions in videos. The most important features for action recognition are the spatiotemporal and motion features

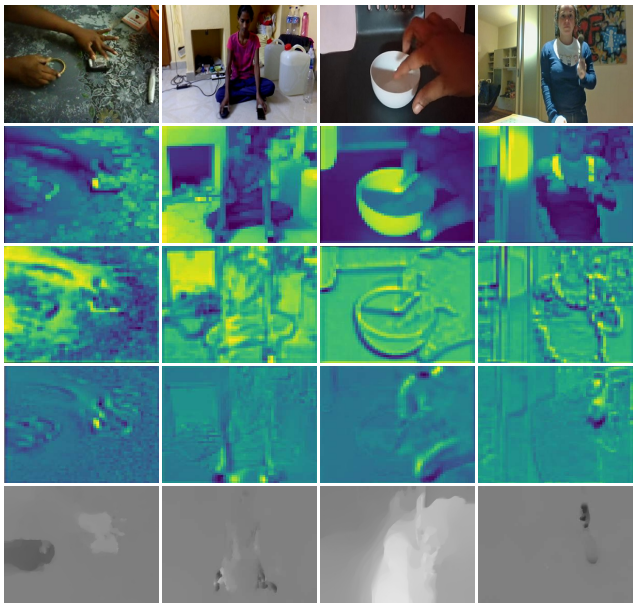

Figure 1. Feature visualization of STM block. First row is the input frames. Second row is the input feature maps of Conv2_1 block. Third row is the output spatiotemporal feature maps of CSTM. The fourth row is the output motion feature maps of CMM. The last row is the optical flow extracted by TV-L1.

where the former encodes the relationship of spatial features from different timestamps while the latter presents motion features between neighboring frames.

The existing methods for action recognition can be summarized into two categories. The first type is based on two-stream neural networks [10, 33, 36, 9], which consists of an RGB stream with RGB frames as input and a flow stream with optical flow as input. The spatial stream models the appearance features (not spatiotemporal features) without considering the temporal information. The flow stream is usually called as a temporal stream, which is designed to

*The work was done during an internship at SenseTime.
†Corresponding author.

model the temporal cues. However, we argue that it is inaccurate to refer the flow stream as the temporal stream because the optical flow only represent the motion features between the neighboring frames and the structure of this stream is almost the same to the spatial stream with 2D CNN. Therefore, this flow stream lacks of the ability to capture the long-range temporal relationship. Besides, the extraction of optical flow is expensive in both time and space, which limits vast industrial applications in the real world.

The other category is the 3D convolutional networks (3D CNNs) based methods, which is designed to capture the spatiotemporal features[27, 2, 24, 3]. 3D convolution is able to represent the temporal features as well as the spatial features together benefiting from the extended temporal dimension. With stacked 3D convolutions, 3D CNNs can capture long-range temporal relationship. Recently, the optimization of this framework with tremendous parameters becomes popular because of the release of large-scale video datasets such as Kinetics [2]. With the help of pre-training on large-scale video datasets, 3D CNN based methods have achieved superior performance to 2D CNN based methods. However, although 3D CNN can model spatiotemporal information from RGB inputs directly, many methods [29, 2] still integrate an independent optical-flow motion stream to further improve the performance with motion features. Therefore, these two features are complementary to each other in action recognition. Nevertheless, expanding the convolution kernel from 2D to 3D and the two-stream structure will inevitably increase the computing cost by an order of magnitude, which limits its real applications.

Inspired by the above observation, we propose a simple yet effective method referred as STM network, to integrate both SpatioTemporal and Motion features in a unified 2D CNN framework, without any 3D convolution and optical flow pre-calculation. Given an input feature map, we adopt a Channel-wise Spatiotemporal Module (CSTM) to present the spatiotemporal features and a Channel-wise Motion Module (CMM) to encode the motion features. We also insert an identity mapping path to combine them together as a block named STM block. The STM blocks can be easily inserted into existing ResNet [13] architectures by replacing the original residual blocks to form the STM networks with negligible extra parameters. As shown in Fig. 1, we visualize our STM block with CSTM and CMM features. The CSTM has learned the spatiotemporal features which pay more attention on the main object parts of the action interaction compared to the original input features. As for the CMM, it captures the motion features with the distinct edges just like optical flow. The main contributions of our work can be summarized as follows:

- We propose a Channel-wise Spatiotemporal Module (CSTM) and a Channel-wise Motion Module (CMM) to encode the complementary spatiotemporal and motion features in a unified 2D CNN framework.
- A simple yet effective network referred as STM Network is proposed with our STM blocks, which can be inserted into existing ResNet architecture by introducing very limited extra computation cost.
- Extensive experiments demonstrate that by integrating both spatiotemporal and motion features together, our method outperforms the state-of-the-art methods on several public benchmark datasets including Something-Something[11], Kinetics [2], Jester [1], UCF101 [23] and HMDB-51 [17].

2. Related Works

With the great success of deep convolution networks in the computer vision area, a large number of CNN-based methods have been proposed for action recognition and have gradually surpassed the performance of traditional methods [30, 31]. A sequence of advances adopt 2D CNNs as the backbone and classify a video by simply aggregating frame-wise prediction [16]. However, these methods only model the appearance feature of each frame independently while ignore the dynamics between frames, which results in inferior performance when recognizing temporal-related videos. To handle the mentioned drawback, two-stream based methods [10, 33, 36, 3, 9] are introduced by modeling appearance and dynamics separately with two networks and fuse two streams through middle or at last. Among these methods, Simonyan et al. [22] first proposed the two-stream ConvNet architecture with both spatial and temporal networks. Temporal Segment Networks (TSN) [33] proposed a sparse temporal sampling strategy for the two-stream structure and fused the two streams by a weighted average at the end. Feichtenhofer et al. [8, 9] studied the fusion strategies in the middle of the two streams in order to obtain the spatiotemporal features. However, these types of methods mainly suffer from two limitations. First, these methods need pre-compute optical flow, which is expensive in both time and space. Second, the learned feature and final prediction from multiple segments are fused simply using weighted or average sum, making it inferior to temporal-relationship modeling.

Another type of methods tries to learn spatiotemporal features from RGB frames directly with 3D CNN [27, 2, 4, 7, 24]. C3D [27] is the first work to learn spatiotemporal features using deep 3D CNN. However, with tremendous parameters to be optimized and lack of high-quality large-scale datasets, the performance of C3D remains unsatisfactory. I3D [2] inflated the ImageNet pre-trained 2D kernel into 3D to capture spatiotemporal features and modeled motion features with another flow stream. I3D has achieved very competitive performance in benchmark datasets with the help of high-quality large-scale Kinetics dataset and the

two-stream setting. Since 3D CNNs try to learn local correlation along the input channels, STCNet [4] inserted its STC block into 3D ResNet to captures both spatial-channels and temporal-channels correlation information throughout network layers. Slowfast [7] involved a slow path to capture spatial semantics and a fast path to capture motion at fine temporal resolution. Although 3D CNN based methods have achieved state-of-the-art performance, they still suffer from heavy computation, making it hard to deploy in real-world applications.

To handle the heavy computation of 3D CNNs, several methods are proposed to find the trade-off between precision and speed [28, 37, 42, 41, 25, 20]. Tran et al. [28] and Xie et al. [37] discussed several forms of spatiotemporal convolutions including employing 3D convolution in early layers and 2D convolution in deeper layers (bottom-heavy) or reversed the combinations (top-heavy). P3D [20] and R(2+1)D [28] tried to reduce the cost of 3D convolution by decomposing it into 2D spatial convolution and 1D temporal convolution. TSM [19] further introduced the temporal convolution by shifting part of the channels along the temporal dimension. Our proposed CSTM branch is similar to these methods in the mean of learning spatiotemporal features, while we employ channel-wise 1D convolution to capture different temporal relationship for different channels. Though these methods are successful in balancing the heavy computation of 3D CNNs, they inevitably need the help of two-stream networks with a flow stream to incorporate the motion features to obtain their best performance. Motion information is the key difference between video-based recognition and image-based recognition task. However, calculating optical flow with TV-L1 method [38] is expensive in both time and space. Recently many approaches have been proposed to estimate optical flow with CNN [5, 14, 6, 21] or explored alternatives of optical flow [33, 39, 26, 18]. TSN frameworks [33] involved RGB difference between two frames to represent motion in videos. Zhao et al. [39] used cost volume processing to model apparent motion. Optical Flow guided Feature (OFF) [26] contains a set of operators including sobel and element-wise subtraction for OFF generation. MFNet [18] adopted five fixed motion filters as a motion block to find feature-level temporal features between two adjacent time steps. Our proposed CMM branch is also designed for finding better yet lightweight alternative motion representation. The main difference is that we learn different motion features for different channels for every two adjacent time steps.

3. Approach

In this section, we will introduce the technical details of our approach. First, we will describe the proposed CSTM and CMM to show how to perform the channel-wise spatiotemporal fusion and extract the feature-level motion information, respectively. Afterward, we will present the combination of these two modules to assemble them as a building block that can be inserted into existing ResNet architecture to form our STM network.

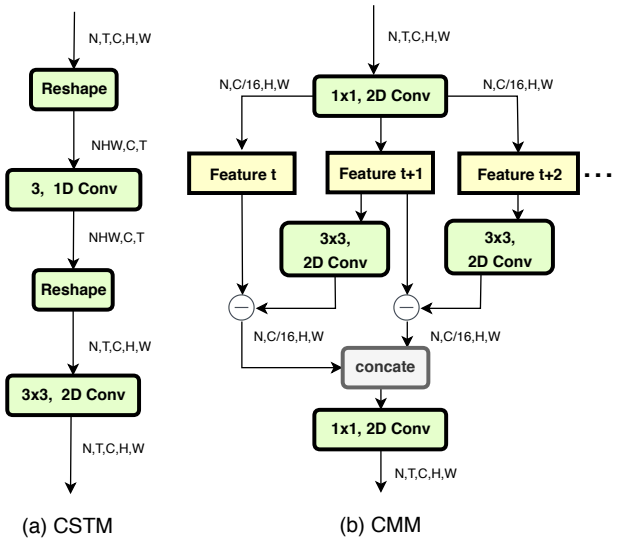

Figure 2. Architecture of Channel-wise SpatioTemporal Module and Channel-wise Motion Module. The feature maps are shown as the shape of their tensors. "⊖" denotes element-wise subtraction.

3.1. Channel-wise SpatioTemporal Module

The CSTM is designed for efficient spatial and temporal modeling. By introducing very limited extra computing cost, CSTM extracts rich spatiotemporal features, which can significantly boost the performance of temporal-related action recognition. As illustrated in Fig. 2(a), given an input feature map $\mathbf{F} \in \mathbb{R}^{N\times T\times C\times H\times W}$, we first reshape $\mathbf{F}$ as: $\mathbf{F} \rightarrow \mathbf{F}^* \in \mathbb{R}^{NHW\times C\times T}$ and then apply the channel-wise 1D convolution on the T dimension to fuse the temporal information. There are mainly two advantages to adopt the channel-wise convolution rather than the ordinary convolution. Firstly, for the feature map $\mathbf{F}^*$, the semantic information of different channels is typically different. We claim that the combination of temporal information for different channels should be different. Thus the channel-wise convolution is adopted to learn independent kernels for each channel. Secondly, compared to the ordinary convolution, the computation cost can be reduced by a factor of G where G is the number of groups. In our settings, G is equal to the number of input channels. Formally, the channel-wise temporal fusion operation can be formulated as:

$$\mathbf{G}_{c,t} = \sum_i \mathbf{K}_i^c \mathbf{F}^*_{c,t+i} \tag{1}$$

where $\mathbf{K}_i^c$ are temporal combination kernel weights belong to channel c and i is the index of temporal kernel, $\mathbf{F}^*_{c,t+i}$

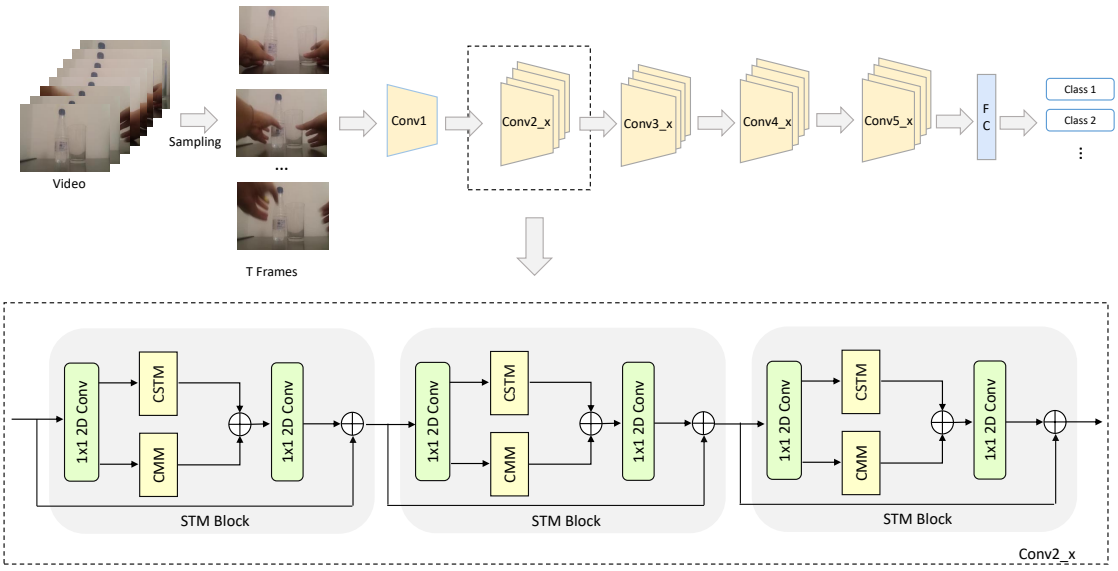

Figure 3. The overall architecture of STM network. The input video is first split into N segments equally and then one frame from each segment is sampled. We adopt 2D ResNet-50 as backbone and replace all residual blocks with STM blocks. No temporal dimension reduction performed apart from the last score fusion stage.

is the input feature sequence and $\mathbf{G}_{c,t}$ is the updated version of the channel-wise temporal fusion features. Here the temporal kernel size is set to 3 thus $i \in [-1, 1]$. Next we will reshape the $\mathbf{G}$ to the original input shape (i.e. $[N, T, C, H, W]$) and model local-spatial information via 2D convolution whose kernel size is 3x3.

We visualize the output feature maps of CSTM to help understand this module in Fig. 1. Compare the features in the second row to the third row, we can find that the CSTM has learned the spatiotemporal features which pay more attention in the main part of the actions such as the hands in the first column while the background features are weak.

3.2. Channel-wise Motion Module

As discovered in [29, 2], apart from the spatiotemporal features directly learned by 3D CNN from the RGB stream, the performance can still be greatly improved by including an optical-flow motion stream. Therefore, apart from the CSTM, we propose a lightweight Channel-wise Motion Module (CMM) to extract feature-level motion patterns between adjacent frames. Note that our aim is to find the motion representation that can help to recognize actions in an efficient way rather than accurate motion information (optical flow) between two frames. Therefore, we will only use the RGB frames and not involve any pre-computed optical flow.

Given the input feature maps $\mathbf{F} \in \mathbb{R}^{N \times T \times C \times H \times W}$, we will first leverage a 1x1 convolution layer to reduce the spatial channels by a factor of r to ease the computing cost, which is setting to 16 in our experiments. Then we generate feature-level motion information from every two consecutive feature maps. Taking $\mathbf{F}_t$ and $\mathbf{F}_{t+1}$ for example, we first apply 2D channel-wise convolution to $\mathbf{F}_{t+1}$ and then subtracts from $\mathbf{F}_t$ to obtain the approximate motion representation $\mathbf{H}_t$:

$$\mathbf{H}_t = \sum_{i,j} \mathbf{K}^c_{i,j} \mathbf{F}_{t+1,c,h+i,w+j} - \mathbf{F}_t \tag{2}$$

where c, t, h, w denote spatial, temporal channel and two spatial dimensions of the feature map respectively and $\mathbf{K}^c_{i,j}$ denotes the c-th motion filter with the subscripts i, j denote the spatial indices of the kernel. Here the kernel size is set to 3×3 thus $i, j \in [-1, 1]$.

As shown in Fig. 2(b), we perform the proposed CMM to every two adjacent feature maps over the temporal dimension, i.e., $\mathbf{F}_t$ and $\mathbf{F}_{t+1}$, $\mathbf{F}_{t+1}$ and $\mathbf{F}_{t+2}$, etc. Therefore, the CMM will produce $T - 1$ motion representations. To keep the temporal size compatible with the input feature maps, we simply use zero to represent the motion information of the last time step and then concatenate them together over the temporal dimension. In the end, another 1x1 2D convolution layer is applied to restore the number of channels to C.

We find that the proposed CMM can boost the performance of the whole model even though the design is quite simple, which proves that the motion features obtained with CMM are complementary to the spatiotemporal features from CSTM. We visualize the motion features learned by CMM in Fig. 1. From which we can see that compared to the output of CSTM, CMM is able to capture the motion

features with the distinct edges just like optical flows.

3.3. STM Network

In order to keep the framework effective yet lightweight, we combine the proposed CSTM and CMM together to build an STM block that can encode spatiotemporal and motion features together and can be easily inserted into the existing ResNet architectures. The overall design of the STM block is illustrated in the bottom half of Fig. 3. In this STM block, the first 1x1 2D convolution layer is responsible for reducing the channel dimensions. The compressed feature maps are then passed through the CSTM and CMM to extract spatiotemporal and motion features respectively. Typically, there are two kinds of ways to aggregate different type of information: summation and concatenation. We experimentally found that summation works better than concatenation to fuse these two modules. Therefore, an element-wise sum operation is applied after the CSTM and CMM to aggregate the information. Then another 1x1 2D convolution layer is applied to restore the channel dimensions. Similar to the ordinary residual block, we also add a parameter-free identity shortcut from the input to the output.

Because the proposed STM block is compatible with the ordinary residual block, we can simply insert it into any existing ResNet architectures to form our STM network with very limited extra computation cost. We illustrate the overall architecture of STM network in the top half of Figure 3. The STM network is a 2D convolutional network which avoids any 3D convolution and pre-computing optical flow. Unless specified, we choose the 2D ResNet-50 [13] as our backbone for its tradeoff between the accuracy and speed. We replace all residual blocks with the proposed STM blocks.

4. Experiments

In this section, we first introduce the datasets and the implementation details of our proposed approach. Then we perform extensive experiments to demonstrate that the proposed STM outperforms all the state-of-the-art methods on both temporal-related datasets (i.e., Something-Something v1 & v2 and Jester) and scene-related datasets (i.e., Kinetics-400, UCF-101, and HMDB-51). The baseline method in our experiments is Temporal Segment Networks (TSN) [33] where we replace the backbone to ResNet-50 for fair comparisons. We also conduct abundant ablation studies with Something-Something v1 to analyze the effectiveness of our method. Finally, we give runtime analyses to show the efficiency of STM compare with state-of-the-art methods.

4.1. Datasets

We evaluate the performance of the proposed STM on several public action recognition datasets. We classify these datasets into two categories: (1) temporal-related datasets, including Something-Something v1 & v2 [11] and Jester [1]. For these datasets, temporal motion interaction of objects is the key to action understanding. Most of the actions cannot be recognized without considering the temporal relationship; (2) scene-related datasets, including Kinetics-400 [2], UCF-101 [23] and HMDB-51 [17] where the background information contributes a lot for determining the action label in most of the videos. Temporal relation is not as important as it in the first group of datasets. We also give examples in Figure 4 to show the difference between them. Since our method is designed for effective spatiotemporal fusion and motion information extraction, we mainly focus on those temporal-related datasets. Nevertheless, for those scene-related datasets, our method also achieves competitive results.

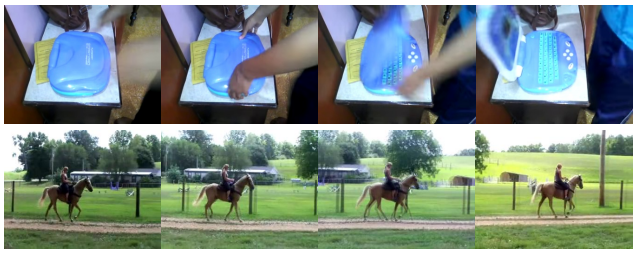

Figure 4. Difference between temporal-related datasets and scene-related datasets. Top: action for which temporal feature matters. Reversing the order of frames gives the opposite label (opening something vs closing something). Bottom: action for which scene feature matters. Only one frame can predict label (horse riding).

4.2. Implementation Details

Training. We train our STM network with the same strategy as mentioned in TSN [33]. Given an input video, we first divide it into T segments of equal durations in order to conduct long-range temporal structure modeling. Then, we randomly sample one frame from each segment to obtain the input sequence with T frames. The size of the short side of these frames is fixed to 256. Meanwhile, corner cropping and scale-jittering are applied for data argumentation. Finally, we resize the cropped regions to 224×224 for network training. Therefore, the input size of the network is $N \times T \times 3 \times 224 \times 224$, where N is the batch size and T is the number of the sampled frames per video. In our experiments, T is set to 8 or 16.

We train our model with 8 GTX 1080TI GPUs and each GPU processes a mini-batch of 8 video clips (when $T = 8$) or 4 video clips (when $T = 16$). For Kinetics, Something-Something v1 & v2 and Jester, we start with a learning rate of 0.01 and reduce it by a factor of 10 at 30,40,45 epochs and stop at 50 epochs. For these large-scale datasets, we only use the ImageNet pre-trained model as initialization.

Table 1. Performance of the STM on the Something-Something v1 and v2 datasets compared with the state-of-the-art methods.

Method	Backbone	Flow	Pretrain	Frame	Something-Something v1			Something-Something v2			
					top-1 val	top-5 val	top-1 test	top-1 val	top-5 val	top-1 test	top-5 test
S3D-G [37]	Inception		ImageNet	64	48.2	78.7	42.0	-	-	-	-
ECO [42]	BNInception+ 3D ResNet-18		Kinetics	8	39.6	-	-	-	-	-	-
ECO [42]				16	41.4	-	-	-	-	-	-
ECO$_{EN}Lite$ [42]				92	46.4	-	42.3	-	-	-	-
ECO$_{EN}Lite$ Two-Stream [42]		✓		92+92	49.5	-	43.9	-	-	-	-
I3D [2]	3D ResNet-50		Kinetics	32	41.6	72.2	-	-	-	-	-
I3D+GCN [2]				32	43.4	75.1	-	-	-	-	-
TSN [33]	ResNet-50		Kinetics	8	19.7	46.6	-	27.8	57.6	-	-
				16	19.9	47.3	-	30.0	60.5	-	-
TRN Multiscale [40]	BNInception		ImageNet	8	34.4	-	33.6	48.8	77.64	50.9	79.3
TRN Two-Stream [40]		✓		8+8	42.0	-	40.7	55.5	83.1	56.2	83.2
MFNet-C101 [18]	ResNet-101		Scratch	10	43.9	73.1	37.5	-	-	-	-
TSM [19]	ResNet-50		Kinetics	16	44.8	74.5	-	58.7	84.8	59.9	85.9
TSM Two-Stream [19]		✓		16+8	49.6	79.0	**46.1**	63.5	88.6	**63.7**	89.5
STM	ResNet-50		ImageNet	8	49.2	79.3	-	62.3	88.8	61.3	88.4
				16	**50.7**	**80.4**	43.1	**64.2**	**89.8**	63.5	**89.6**

Table 2. Performance of the STM on the Jester compared with the state-of-the-art methods.

Method	Backbone	Frame	Top-1	Top-5
TSN [33]	ResNet-50	8	81.0	99.0
		16	82.3	99.2
TRN-Multiscale [40]	BNInception	8	95.3	-
MFNet-C50 [18]	ResNet-50	7	96.1	99.7
TSM [19]	ResNet-50	8	94.4	99.7
		16	95.3	99.8
STM	ResNet-50	8	96.6	99.9
		16	**96.7**	**99.9**

Table 3. Performance of the STM on the Kinetics-400 dataset compared with the state-of-the-art methods.

Method	Backbone	Flow	Top-1	Top-5
STC [4]	ResNext101		68.7	88.5
ARTNet [32]	ResNet-18		69.2	88.3
ECO [42]	BNInception +3D ResNet-18		70.7	89.4
S3D [37]	Inception		72.2	90.6
I3D RGB [2]	3D Inception-v1		71.1	89.3
I3D Two-Stream [2]		✓	**74.2**	91.3
StNet [12]	ResNet-101		71.4	-
Disentangling [39]	BNInception		71.5	89.9
R(2+1)D RGB [28]	ResNet-34		72.0	90.0
R(2+1)D Two-Stream [28]		✓	73.9	90.9
TSM [19]	ResNet-50		72.5	90.7
TSN RGB [33]	BNInception		69.1	88.7
TSN Two-Stream [33]		✓	73.9	91.1
STM	ResNet-50		73.7	**91.6**

For the temporal channel-wise 1D convolution in CSTM, first quarter of channels are initialized to [1,0,0], last quarter of channels are initialized to [0,0,1] and other half are [0,1,0]. All parameters in CMM are randomly initialized. For UCF-101 and HMDB-51, we use Kinetics pre-trained model as initialization and start training with a learning rate of 0.001 for 25 epochs. The learning rate is decayed by a factor 10 every 15 epochs. We use mini-batch SGD as optimizer with a momentum of 0.9 and a weight decay of 5e-4. Different from [33], we enable all the BatchNorm layers [15] during training.

Inference. Following [34, 7], we first scale the shorter spatial side to 256 pixels and take three crops of 256×256 to cover the spatial dimensions and then resize them to 224×224. For the temporal domain, we randomly sample 10 times from the full-length video and compute the softmax scores individually. The final prediction is the averaged softmax scores of all clips.

4.3. Results on Temporal-Related Datasets

In this section, we compare our approach with the state-of-the-art methods on temporal-related datasets including Something-Something v1 & v2 and Jester. Something-Something v1 is a large collection of densely-labeled video clips which shows basic human interactions with daily objects. This dataset contains 174 classes with 108,499 videos. Something-Something v2 is an updated version of v1 with more videos (220,847 in total) and greatly reduced label noise. Jester is a crowd-acted video dataset for generic human hand gestures recognition, which contains 27 classes with 148,092 videos.

Table 1 lists the results of our method compared with the state-of-the-art on Something-Something v1 and v2. The results of the baseline method TSN are relatively low compared with other methods, which demonstrates the importance of temporal modeling for these temporal-related datasets. Compared with the baseline method, our STM network gains 29.5% and 30.8% top-1 accuracy improvement with 8 and 16 frames inputs respectively on Something-Something v1. On Something-Something v2, STM also gains 34.5% and 34.2% improvement compared to TSN. The rest part of Table 1 shows the other state-of-the-art methods. These methods can be classified into two types as shown in the two parts of Table 1. The upper part presents the 3D CNN based methods, including S3D-G [37], ECO [42] and I3D+GCN models [35]. The lower part is 2D CNN based methods, including TRN [40], MFNet [18] and TSM [19]. It is clear that even STM with 8 RGB frames

Table 4. Performance of the STM on UCF-101 and HMDB-51 compared with the state-of-the-art methods.

Method	Backbone	Flow	Pre-train Data	UCF-101	HMDB-51
C3D [27]	3D VGG-11		Sports-1M	82.3	51.6
STC [4]	ResNet101		Kinetics	93.7	66.8
ARTNet with TSN [32]	3D ResNet-18		Kinetics	94.3	70.9
ECO [42]	BNInception+3D ResNet-18		Kinetics	94.8	72.4
I3D RGB [2]	3D Inception-v1		ImageNet+Kinetics	95.1	74.3
I3D two-stream [2]		✓		98.0	80.7
TSN [33]	ResNet-50		ImageNet	86.2	54.7
TSN RGB [33]	BNInception		ImageNet+Kinetics	91.1	-
TSN two-Stream [33]		✓		97.0	-
TSM [19]	ResNet-50		ImageNet+Kinetics	94.5	70.7
StNet [12]	ResNet50		ImageNet+Kinetics	93.5	-
Disentangling [39]	BNInception		ImageNet+Kinetics	95.9	-
STM	ResNet-50		ImageNet+Kinetics	96.2	72.2

as input achieves the state-of-the-art performance compared with other methods, which take more frames and optical flow as input or 3D CNN as the backbone. With 16 frames as input, STM achieves the best performance in the validation sets of both Something-Something v1 and v2, and just a little lower in the top1 accuracy in the test sets, which adopts only 16 RGB frames as input.

Table 2 shows the results on the Jester dataset. Our STM also gains a large improvement compared to the TSN baseline method, and outperforms all the state-of-the-art methods.

4.4. Results on Scene-Related Datasets

We evaluate our STM on three scene-related datasets: Kinetics-400, UCF-101, and HMDB-51 in this section. Kinetics-400 is a large-scale human action video dataset with 400 classes. It contains 236,763 clips for training and 19,095 clips for validation. UCF-101 is a relatively small dataset which contains 101 categories and 13,320 clips in total. HMDB-51 is also a small video dataset with 51 classes and 6766 labeled video clips. For UCF-101 and HMDB-51, we followed [33] to adopt the three training/testing splits for evaluation.

Table 3 summaries the results of STM and other competing methods on the Kinetics-400 dataset. We train STM with 16 frames as input, and the same for evaluation. From the evaluation results, we can draw the following conclusions: (1) Different from the previous temporal-related datasets, most actions of Kinetics can be recognized by scene and objects even with one still frame of videos, therefore the baseline method without any temporal modeling can achieve acceptable accuracy; (2) Though our method is mainly focused on temporal-related actions recognition, STM still achieves very competitive results compare with the state-of-the-art methods. Top-1 accuracy of our method is only 0.5% lower than the two-stream I3D, which involves both 3D convolution and pre-computation optical flow. However, STM outperforms major recently proposed 3D CNN based methods (the upper part of the Table 3) as well as 2D CNN based methods (the lower part of the Table 3) and achieve the best top-5 accuracy compared with all the other method.

We also conduct experiments on the UCF-101 and HMDB-51 to study the generalization ability of learned spatiotemporal and motion representations. We evaluate our method over three splits and report the averaged results in Table 4. First, compared with the ImageNet pre-trained model, Kinetics pre-train can significantly improve the performance on small datasets. Then, compare with the state-of-the-art methods, only two methods, I3D two-stream and TSN two-Stream, performs a little better than ours while both of them utilize optical flow as their extra inputs. However, STM with 16 frames as inputs even outperforms I3D with RGB stream on UCF101, which also uses Kinetics as pre-train data but the 3D CNN leads to much higher computation cost than ours.

4.5. Ablation Studies

In this section, we comprehensively evaluate our proposed STM on Something-Something v1 dataset. All the ablation experiments in this section use 8 RGB frames as inputs.

Impact of two modules. Our proposed two modules can be inserted into a standard ResNet architecture independently. To validate the contributions of each component in the STM block (i.e., CSTM and CMM), we compare the results of the individual module and the combination of both modules in Table 5. We can see that each component contributes to the proposed STM block. CSTM learns channel-wise temporal fusion and brings about 28% top-1 accuracy improvement compared to the baseline method TSN while CMM learns feature-level motion information and brings 24.4% top-1 accuracy improvement. When combining CSTM and CMM together, we can learn richer spatiotemporal and mo-

Model	Top-1	Top-5
TSN	19.7	46.6
CSTM	47.7	77.9
CMM	44.1	74.8
STM	49.2	79.3

Table 5. **Impact of two modules:** Comparison between CSTM, CMM and STM.

Aggregation	Top-1	Top-5
TSN	19.7	46.6
Summation	49.2	79.3
Concatenation	41.8	73.2

Table 6. **Fusion of two modules:** Summation fusion is better.

Stage	STM Blocks	Top-1	Top-5
2	1	38.7	70.1
3	1	40.6	71.6
4	1	41.5	72.6
5	1	41.5	71.8
2-5	4	47.9	78.1
2-5	16	49.2	79.3

Table 7. **Location and number of STM block:** Deeper location and more blocks yeild better performance.

Type	Channel-wise	Ordinary
Top-1 Acc.	47.7	46.9
Param.	23.88M	27.64M
FLOPs	32.93G	40.59G

Table 8. **Type of temporal convolution in CSTM:** Channel-wise temporal convolution yields better performance.

tion features and achieve the best top-1 accuracy, especially, the gain over the baseline is 29.5%.

Fusion of two modules. There are two ways to combine CSTM and CMM: element-wise summation and concatenation. The element-wise summation is parameter-free and easy to implement. For concatenation fusion, we first concatenate outputs of CSTM and CMM over the channel dimension, and the dimension of concatenate features is $2C$. Then a 1x1 convolution is applied to reduce the channels to C. We conduct the experiments to study the two fusion ways as shown in Table 6, though summation aggregation is simple, it still outperforms concatenation by 7.4% at top-1 accuracy and 6.1% at top-5 accuracy.

Location and number of STM block. ResNet-50 architecture can be divided into 6 stages. We refer the conv2_x to conv5_x as stage 2 to stage 5. The first four rows of Table 7 compare the performance of replacing only the first residual block with STM on different stages in ResNet-50, from stage 2 to stage 5, respectively. We conclude from the results that replacing only one residual block already yield significant performance improvement compared to the baseline TSN, which demonstrates the effectiveness of the proposed STM block. One may notice that replacing the STM block at latter stage (e.g., stage 5) yield better accuracy than early stage (e.g., stage 2). One possible reason is that temporal modeling is beneficial more with larger receptive fields which can capture holistic features. We then replace one block for each stage (i.e., replacing four blocks in all) and leads to better results. When replacing all original residual blocks with STM blocks (i.e., 16 blocks in all), our model achieves the best performance.

Type of temporal convolution in CSTM. We choose channel-wise temporal convolution in CSTM to learn temporal combination individually for each channel. We also make comparison with ordinary temporal convolution in CSTM module and the result is shown in Table 8. With channel-wise convolution, we can achieve better performance with few parameters and FLOPs.

4.6. Runtime Analysis

Our STM achieves the new state-of-the-art results on several benchmark datasets compared with other methods.

Table 9. Accuracy and model complexity of STM and other state-of-the-art methods on Something-Something V1 dataset. Single crop STM beats all competing methods with 62 videos per second with 8 frames as input. Measured on a single NVIDIA GTX 1080TI GPU.

Model	Frame	FLOPs	Param.	Speed	Acc.
I3D [2]	64	306G	28.0M	6.4 V/s	41.6
ECO [42]	16	64G	47.5M	46.3 V/s	41.4
TSM [19]	8	32.9G	23.9M	80.4 V/s	43.8
	16	65.8G		40.6 V/s	44.8
STM	8	33.3G	24.0M	62.0 V/s	47.5
	16	66.5G		32.0 V/s	**49.8**

More importantly, it is a unified 2D CNN framework without any time-consuming 3D convolution and optical flow calculations. Table 9 shows the accuracy and model complexity of STM and several state-of-the-art methods on Something-Something v1 dataset. All evaluations are running on one GTX 1080TI GPU. For a fair comparison, we evaluate our method by evenly sampling 8 or 16 frames from a video and then apply the center crop. To evaluate speed, we use a batch size of 16 and ignore the time of data loading. Compared to I3D and ECO, STM achieves approximately 10x and 2x less FLOPs (33.3G vs 306G, 64G) while 5.9% and 6.1% higher accuracy. Compared to $\mathrm{TSM}_{16\mathrm{F}}$, our $\mathrm{STM}_{8\mathrm{F}}$ gains 2.7% higher accuracy with 1.5x faster speed and half FLOPs.

5. Conclusion

In this paper, we presented a simple yet effective network for action recognition by encoding spatiotemporal and motion features together in a unified 2D CNN network. We replace the original residual blocks with STM blocks in ResNet architecture to build the STM network. An STM block contains a CSTM to model channel-wise spatiotemporal feature and a CMM to model channel-wise motion representation together. Without any 3D convolution and pre-calculation optical flow, our STM receives state-of-the-art results on both temporal-related datasets and scene-related datasets with only 1.2% more FLOPs compared to TSN baseline.

References

[1] The 20bn-jester dataset v1. In *https://20bn.com/datasets/jester*.

[2] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017.

[3] Yunpeng Chen, Yannis Kalantidis, Jianshu Li, Shuicheng Yan, and Jiashi Feng. Aˆ 2-nets: Double attention networks. In *Advances in Neural Information Processing Systems*, pages 350–359, 2018.

[4] Ali Diba, Mohsen Fayyaz, Vivek Sharma, M Mahdi Arzani, Rahman Yousefzadeh, Juergen Gall, and Luc Van Gool. Spatio-temporal channel correlation networks for action classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 284–299, 2018.

[5] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2758–2766, 2015.

[6] Lijie Fan, Wenbing Huang, Chuang Gan, Stefano Ermon, Boqing Gong, and Junzhou Huang. End-to-end learning of motion representation for video understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6016–6025, 2018.

[7] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. *arXiv preprint arXiv:1812.03982*, 2018.

[8] Christoph Feichtenhofer, Axel Pinz, and Richard Wildes. Spatiotemporal residual networks for video action recognition. In *Advances in neural information processing systems*, pages 3468–3476, 2016.

[9] Christoph Feichtenhofer, Axel Pinz, and Richard P Wildes. Spatiotemporal multiplier networks for video action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4768–4777, 2017.

[10] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1933–1941, 2016.

[11] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *ICCV*, volume 2, page 8, 2017.

[12] Dongliang He, Zhichao Zhou, Chuang Gan, Fu Li, Xiao Liu, Yandong Li, Liming Wang, and Shilei Wen. Stnet: Local and global spatial-temporal modeling for action recognition. *arXiv preprint arXiv:1811.01549*, 2018.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[14] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2462–2470, 2017.

[15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.

[16] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, 2014.

[17] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. HMDB: a large video database for human motion recognition. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2011.

[18] Myunggi Lee, Seungeui Lee, Sungjoon Son, Gyutae Park, and Nojun Kwak. Motion feature network: Fixed motion filter for action recognition. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 387–403, 2018.

[19] Ji Lin, Chuang Gan, and Song Han. Temporal shift module for efficient video understanding. *arXiv preprint arXiv:1811.08383*, 2018.

[20] Zhaofan Qiu, Ting Yao, and Tao Mei. Learning spatio-temporal representation with pseudo-3d residual networks. In *proceedings of the IEEE International Conference on Computer Vision*, pages 5533–5541, 2017.

[21] Anurag Ranjan and Michael J Black. Optical flow estimation using a spatial pyramid network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4161–4170, 2017.

[22] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014.

[23] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[24] Jonathan C Stroud, David A Ross, Chen Sun, Jia Deng, and Rahul Sukthankar. D3d: Distilled 3d networks for video action recognition. *arXiv preprint arXiv:1812.08249*, 2018.

[25] Lin Sun, Kui Jia, Dit-Yan Yeung, and Bertram E Shi. Human action recognition using factorized spatio-temporal convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4597–4605, 2015.

[26] Shuyang Sun, Zhanghui Kuang, Lu Sheng, Wanli Ouyang, and Wei Zhang. Optical flow guided feature: a fast and robust motion representation for video action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1390–1399, 2018.

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

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

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

[30] Heng Wang, Alexander Kläser, Cordelia Schmid, and Liu Cheng-Lin. Action recognition by dense trajectories. In *CVPR 2011-IEEE Conference on Computer Vision & Pattern Recognition*, pages 3169–3176. IEEE, 2011.

[31] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *Proceedings of the IEEE international conference on computer vision*, pages 3551–3558, 2013.

[32] Limin Wang, Wei Li, Wen Li, and Luc Van Gool. Appearance-and-relation networks for video classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1430–1439, 2018.

[33] 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 *European conference on computer vision*, pages 20–36. Springer, 2016.

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

[35] Xiaolong Wang and Abhinav Gupta. Videos as space-time region graphs. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 399–417, 2018.

[36] Yunbo Wang, Mingsheng Long, Jianmin Wang, and Philip S Yu. Spatiotemporal pyramid network for video action recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1529–1538, 2017.

[37] Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 305–321, 2018.

[38] Christopher Zach, Thomas Pock, and Horst Bischof. A duality based approach for realtime tv-l 1 optical flow. In *Joint pattern recognition symposium*, pages 214–223. Springer, 2007.

[39] Yue Zhao, Yuanjun Xiong, and Dahua Lin. Recognize actions by disentangling components of dynamics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6566–6575, 2018.

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

[41] Yizhou Zhou, Xiaoyan Sun, Zheng-Jun Zha, and Wenjun Zeng. Mict: Mixed 3d/2d convolutional tube for human action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 449–458, 2018.

[42] Mohammadreza Zolfaghari, Kamaljeet Singh, and Thomas Brox. Eco: Efficient convolutional network for online video understanding. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 695–712, 2018.