Motion Feature Network: Fixed Motion Filter for Action Recognition

Myunggi Lee[1,2,*], Seungeui Lee[1,*], Sungjoon Son[1,2], Gyutae Park[1,2], and Nojun Kwak[1]

[1] Seoul National University, Seoul, South Korea
{myunggi89, dehlix, sjson, pgt4861, nojunk}@snu.ac.kr
[2] V.DO Inc., Suwon, Korea

Abstract. Spatio-temporal representations in frame sequences play an important role in the task of action recognition. Previously, a method of using optical flow as a temporal information in combination with a set of RGB images that contain spatial information has shown great performance enhancement in the action recognition tasks. However, it has an expensive computational cost and requires two-stream (RGB and optical flow) framework. In this paper, we propose MFNet (Motion Feature Network) containing motion blocks which make it possible to encode spatio-temporal information between adjacent frames in a unified network that can be trained end-to-end. The motion block can be attached to any existing CNN-based action recognition frameworks with only a small additional cost. We evaluated our network on two of the action recognition datasets (Jester and Something-Something) and achieved competitive performances for both datasets by training the networks from scratch.

Keywords: action recognition · motion filter · MFNet · spatio-temporal representation

1 Introduction

Convolutional neural networks (CNNs) [17] are originally designed to represent static appearances of visual scenes well. However, it has a limitation if the underlying structure is characterized by sequential and temporal relations. In particular, since recognizing human behavior in a video requires both spatial appearance and temporal motion as important cues, many previous researches have utilized various modalities that can capture motion information such as optical flow [33] and RGBdiff (temporal difference in consecutive RGB frames) [33]. Methods based on two-stream [33, 21, 7] and 3D convolutions [28, 2] utilizing these input modalities achieve state-of-the-art performances in the field of action recognition. However, even though optical flow is a widely utilized modality that provides short-term temporal information, it takes a lot of time to generate. Likewise, 3D-kernel-based methods such as 3D ConvNets also require heavy computational burden with high memory requirements.

* M. Lee and S. Lee equally contributed the paper. This work was supported by the ICT R&D program of MSIP/IITP, Korean Government (2017-0-00306)

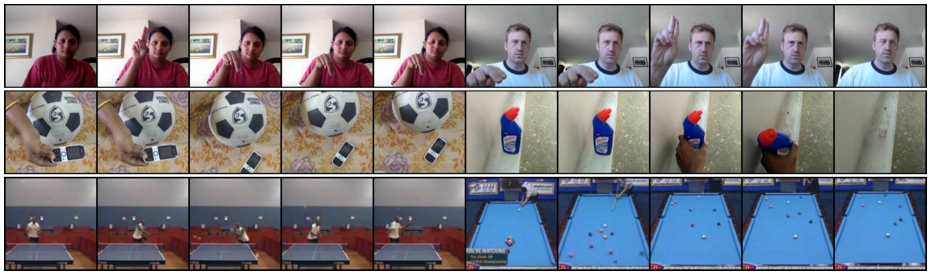


Fig. 1. Some examples of action classes in the three action recognition datasets, Jester (top), Something-Something (middle), and UCF101 (bottom). – top: left *'Sliding Two Fingers Down'*, right *'Sliding Two Fingers Up'*, middle: left *'Dropping something in front of something'*, right *'Removing something, revealing something behind'*, bottom: left *'TableTennisShot'*, right *'Billiards'*. Due to ambiguity of symmetrical pair classes/actions, static images only are not enough to recognize correct labels without sequential information in the former two datasets. However, in case of the bottom UCF101 image frames, the action class can be recognized with only spatial context (*e.g.* background and objects) from a single image.

In our view, most previous labeled action recognition datasets such as UCF101 [24], HMDB51 [16], Sports-1M [13] and THUMOS [12] provide highly abstract concepts of human behavior. Therefore they can be mostly recognized without the help of temporal relations of sequential frames. For example, the *'Billiard'* and *'TableTennisShot'* in UCF101 can be easily recognizable by just seeing one frame as shown in the third row of Fig. 1. Unlike these datasets, Jester [1] and Something-Something [8] include more detailed physical aspects of actions and scenes. The appearance information has a very limited usefulness in classifying actions for these datasets. Also, visual objects in the scenes that mainly provide shape information are less important for the purpose of recognizing actions on these datasets. In particular, the Something-Something dataset has little correlation between the object and the action class, as its name implies. The first two rows of Fig. 1 show some examples of these datasets. As shown in Figure 1, it is difficult to classify the action class with only one image. Also, even if there are multiple images, the action class can be changed according to the temporal order. Thus, it can be easily confused when using the conventional static feature extractors. Therefore, the ability to extract the temporal relationship between consecutive frames is important to classify human behavior in these datasets.

To solve these issues, we introduce a unified model which is named as the Motion Feature Network (MFNet). MFNet contains specially designed motion blocks which represent spatio-temporal relationships from only RGB frames. Because it extracts temporal information using only RGB, pre-computation time that is typically needed to compute optical flow is not needed compared with the existing optical flow-based approaches. Also, because MFNet is based on a 2D CNN architecture, it has fewer parameters compared to its 3D counterparts.

We perform experiments to verify our model's ability to extract spatio-temporal features on a couple of publicly available action recognition datasets. In these datasets, each video label is closely related to the sequential relationships among frames. MFNet trained using only RGB frames significantly outperforms previous methods. Thus, MFNet can be used as a good solution for an action classification task in videos consisting of sequential relationships of detailed physical entities. We also conduct ablation studies to understand properties of MFNets in more detail.

The rest of this paper is organized as follows. Some related works for action recognition tasks are discussed in Section 2. Then in Section 3, we introduce our proposed MFNet architecture in detail. After that, experimental results with ablation studies are presented and analyzed in Section 4. Finally, the paper is concluded in Section 5.

2 Related Works

With the great success of CNNs on various computer vision tasks, a growing number of studies have tried to utilize deeply learned features for action recognition in video datasets. Especially, as the consecutive frames of input data imply sequential contexts, temporal information as well as spatial information is an important cue for classification tasks. There have been several approaches to extract these spatio-temporal features on action recognition problems.

One popular way to learn spatio-temporal features is using 3D convolution and 3D pooling hierarchically [28, 36, 29, 9, 6]. In this approach, they usually stack continuous frames of a video clip and feed them into the network. The 3D convolutions have enough capacity to encode spatio-temporal information on densely sampled frames but are inefficient in terms of computational cost. Furthermore, the number of parameters to be optimized are relatively large compared to other approaches. Thus, it is difficult to train on small datasets, such as UCF101 [24] and HMDB51 [15]. In order to overcome these issues, Carreira *et al.* [2] introduced a new large dataset named Kinetics[14], which facilitates training 3D models. They also suggest inflating 3D convolution filters from 2D convolution filters to bootstrap parameters from the pre-trained ImageNet [4] models. It achieves state-of-the-art performances in action recognition tasks.

Another famous approach is the two-stream-based method proposed by Simonyan *et al.* [22]. It encodes two kinds of modalities which are raw pixels of an image and the optical flow extracted from two consecutive raw image frames. It predicts action classes by averaging the predictions from both a single RGB frame and a stack of externally computed multiple optical flow frames. A large amount of follow up studies[32, 18, 35] to improve the performance of action recognition has been proposed based on the two-stream framework [33, 21, 7]. As an extension to the previous two-stream method, Wang *et al.* [33] proposed the temporal segment network. It samples image frames and optical flow frames on different time segments over the entire video sequences instead of short snippets, then it trains RGB frames and optical flow frames independently. At inference time, it

accumulates the results to predict an activity class. While it brings a significant improvement over traditional methods [3, 30, 31], it still relies on pre-computed optical flows which are computationally expensive.

In order to replace the role of hand-crafted optical flow, there have been some works feeding frames similar to optical flow as inputs to the convolutional networks [33, 36]. Another line of works use optical flow only in training phase as ground-truth [20, 38]. They trained a network that reconstructs optical flow images from raw images and provide the estimated optical flow information to the action recognition network. Recently, Sun *et al.* [26] proposed a method of optical-flow-guided features. It extracts motion representation using two sets of features from adjacent frames by separately applying temporal subtraction (temporal features) and Sobel filters (spatial features). Our proposed method is highly related to this work. The differences are that we feedforward spatial and temporal features in a unified network instead of separating two features apart. Thus, it is possible to train the proposed MFNet in an end-to-end manner.

3 Model

In this section, we first introduce the overall architecture of the proposed MFNet and then give a detailed description of 'motion filter' and 'motion block' which constitute MFNet. We provide several instantiations of motion filter and motion block to explain the intuition behind it.

3.1 Motion Feature Network

The proposed architecture of MFNet is illustrated in Figure 2. We construct our architecture based on *temporal segment network* (TSN) [33] which works on a sequence of K snippets sampled from the entire video. Our network is composed of two major components. One is *appearance block* which encodes the spatial information. This can be any of the architectures used in image classification tasks. In our experiments, we use ResNet [10] as our backbone network for appearance blocks. Another component is *motion block* which encodes temporal information. To model the motion representation, it takes two consecutive feature maps of the corresponding consecutive frames from the same hierarchy[3] as inputs and then extracts the temporal information using a set of fixed motion filters which will be described in the next subsection. The extracted spatial and temporal features in each hierarchy should be properly propagated to the next hierarchy. To fully utilize two types of information, we provide several schemes to accumulate them for the next hierarchy.

3.2 Motion Representation

To capture the motion representation, one of the commonly used approaches in action recognition is using optical flow as inputs to a CNN. Despite its important

[3] We use the term *hierarchy* to represent the level of abstraction. A layer or a block of layers can correspond to a hierarchy.

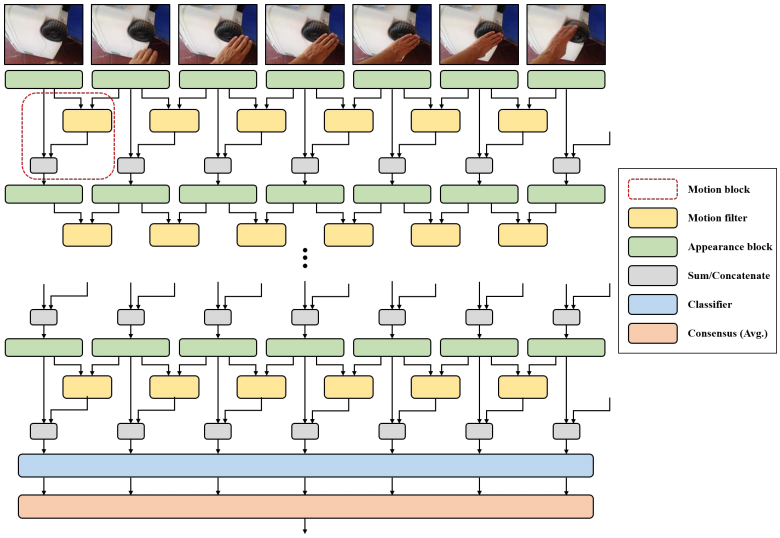


Fig. 2. The overall architecture of MFNet. The proposed network is composed of appearance blocks and motion blocks which encode spatial and temporal information. A motion block takes two consecutive feature maps from respective appearance blocks and extracts spatio-temporal information with the proposed fixed motion filters. The accumulated feature maps from the appearance blocks and motion blocks are used as an input to the next layer. This figure shows the case of $K = 7$.

role in the action recognition tasks, optical flow is computationally expensive in practice. In order to replace the role of optical flow and to extract temporal features, we propose motion filters which have a close relationship with the optical flow.

Approximation of Optical Flow To approximate the feature-level optical flow hierarchically, we propose a modular structure named motion filter. Typically, the brightness consistency constraint of optical flow is defined as follows:

$$I(x + \Delta x, y + \Delta y, t + \Delta t) = I(x, y, t), \tag{1}$$

where $I(x, y, t)$ denotes the pixel value at the location (x, y) of a frame at time t. Here, Δx and Δy denote the spatial displacement in horizontal and vertical axis respectively. The optical flow $(\Delta x, \Delta y)$ that meets (1) is calculated between two consecutive image frames at time t and $t + \Delta t$ at every location of an image.

Originally, solving an optical flow problem is to find the optimal solution $(\Delta x^*, \Delta y^*)$ through an optimization technique. However, it is hard to solve (1) directly without additional constraints such as spatial or temporal smoothness assumptions. Also, it takes much time to obtain a dense (pixelwise) optical flow.

In this paper, the primary goal is to find the temporal features derived from optical flow to help classifying action recognition rather than finding the opti-

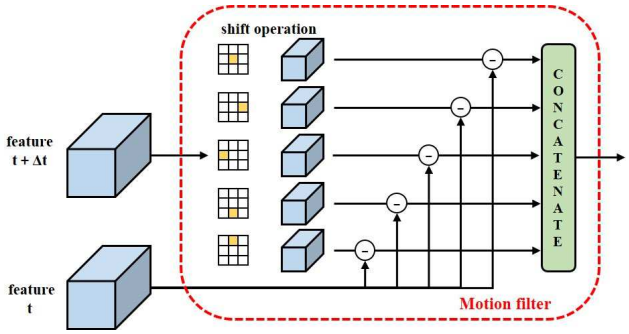


Fig. 3. Motion filter. Motion filter generates spatio-temporal features from two consecutive feature maps. Feature map at time $t + \Delta t$ is shifted by a predefined set of fixed directions and each of them is subtracted from the feature map at time t. With concatenation of features from all directions, motion filter can represent spatio-temporal information.

mal solution to optical flow. Thus, we extend (1) to feature space by replacing an image $I(x, y, t)$ with the corresponding feature maps $F(x, y, t)$ and define a residual features R as follows:

$$R_l(x, y, \Delta t) = F_l(x + \Delta x, y + \Delta y, t + \Delta t) - F_l(x, y, t), \tag{2}$$

where l denotes the index of the layer or hierarchy, F_l is the l-th feature maps from the basic network. R is the residual features produced by two features from the same layer l. Given Δx and Δy, the residual features R can be easily calculated by subtracting two adjacent features at time t and $t + \Delta t$. To fully utilize optical flow constraints in feature level, R tends to have lower absolute intensity. As searching for the lowest absolute value in each location of feature map is trivial but time-consuming, we design a set of predefined fixed directions $\mathbb{D} = \{(\Delta x, \Delta y)\}$ to restrict the search space. For convenience, in our implementation, we restrict $\Delta x, \Delta y \in \{0, \pm 1\}$ and $|\Delta x| + |\Delta y| \leq 1$. Shifting one pixel along each spatial dimension in the image space is responsible for capturing a small amount of optical flow (*i.e.* small movement), while one pixel in the feature space at a higher hierarchy of a CNN can capture larger optical flow (*i.e.* large movement) as it looks at a larger receptive field.

Motion Filter The motion filter is a modular structure calculated by two feature maps extracted from shared networks feed-forwarded by two consecutive frames as inputs. As shown in Figure 3, the motion filter takes features $F_l(t)$ and $F_l(t+\Delta t)$ at time t and $t+\Delta t$ as inputs. The predefined set of directions $\mathbb{D}$ is only applied to the features at time $t + \Delta t$ as illustrated Figure 3. We follow the shift operation proposed in [34]. It moves each channel of its input tensor in a different spatial direction $\delta \triangleq (\Delta x, \Delta y) \in \mathbb{D}$. This can be alternatively done with widely used depth-wise convolution, whose kernel size is determined by the maximum

value of Δx and Δy in $\mathbb{D}$. For example, on our condition, $\Delta x, \Delta y \in \{0, \pm 1\}$, we can implement with 3×3 kernels as shown in Figure 3. Formally, the shift operation can be formulated as:

$$G^{\delta}_{k,l,m} = \sum_{i,j} K^{\delta}_{i,j} F_{k+\hat{i},l+\hat{j},m}, \tag{3}$$

$$K^{\delta}_{i,j} = \begin{cases} 1 & \text{if } i = \Delta x \text{ and } j = \Delta y, \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

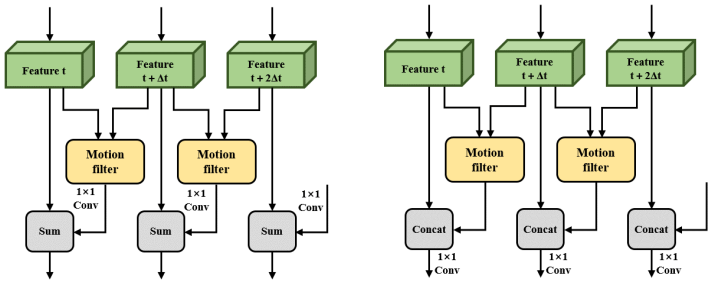
Here, the subscript indicates the index of a matrix or a tensor, $\delta \triangleq (\Delta x, \Delta y) \in \mathbb{D}$ is a displacement vector, $F \in \mathbb{R}^{W \times H \times C}$ is the input tensor and $\hat{i} = i - \lfloor W/2 \rfloor$, $\hat{j} = j - \lfloor H/2 \rfloor$ are the re-centered spatial indices ($\lfloor \cdot \rfloor$ is the floor operation). The indices k, l and i, j are those along spatial dimensions and m is a channel-wise index. We get a set $\mathbb{G}=\{G^{\delta}_{t+\Delta t} | \delta \in \mathbb{D}\}$, where $G^{\delta}_{t+\Delta t}$ represents the shifted feature map by an amount of δ at time $t + \Delta t$. Then, each of them is subtracted by F_t[4]. Because the concatenated feature map is constructed by temporal subtraction on top of the spatially shifted features, the feature map contains spatio-temporal information suitable for action recognition. As mentioned in Section 2, this is quite different from optical-flow-guided features in [26] which use two types of feature maps obtained by temporal subtraction and spatial Sobel filters. Also, it is distinct from 'subtractive correlation layer' in [5] with respect to the implementation and the goal. 'Subtractive correlation layer' is utilized to find correspondences for better reconstruction, while, the proposed motion filter is aimed to encode directional information between two feature maps via learnable parameters.

3.3 Motion Block

As mentioned above, the motion filter is a modular structure which can be adopted to any intermediate layers of two appearance blocks consecutive in time. In order to propagate spatio-temporal information properly, we provide several building blocks. Inspired by the recent success of residual block used in *residual networks* (ResNet) in many challenging image recognition tasks, we develop a new building block named motion block to propagate spatio-temporal information between two adjacent appearance blocks into deeper layers.

Element-wise Sum A simple and direct way to aggregate two different characteristics of information is the element-wise sum operation. As illustrated in Figure 4(a), a set of motion features $R^{\delta}_t \triangleq F_t - G^{\delta}_{t+\Delta t} \in \mathbb{R}^{W \times H \times C}$, $\delta \in \mathbb{D}$, generated by motion filter are concatenated along channel dimension to produce a tensor $M_t = [R^{\delta_1}_t | R^{\delta_2}_t | \cdots | R^{\delta_S}_t] \in \mathbb{R}^{W \times H \times N}$, where $[\cdot|\cdot]$ denotes a concatenation

[4] For convenience, here, we use the notation F_t and $G_{t+\Delta t}$ instead of $F(t)$ and $G(t + \Delta t)$. The meaning of a subscript will be obvious in the context.



(a) Element-wise sum (b) Concatenation

Fig. 4. Two ways to aggregate spatial and temporal information from appearance block and motion filter.

operation, $N = S \times C$ and S is the number of the predefined directions in $\mathbb{D}$. It is further compressed by 1×1 convolution filters to produce output $\hat{M}_t$ with the same dimension as F_t. Finally, the features from the appearance block F_t and those from the motion filters $\hat{M}_t$ are summed up to produce inputs to the next hierarchy.

Concatenation Another popular way to combine the appearance and the motion features is calculated by the concatenation operation. In this paper, the motion features M_t mentioned above are directly concatenated with each of the appearance features F_t as depicted in Figure 4(b). A set of 1×1 convolution filters is also exploited to encode spatial and temporal information after the concatenation. The 1×1 convolution reduces the channel dimension as we desire. It also implicitly encodes spatio-temporal features to find the relationship between two different types of features: appearance and motion features.

4 Experiments

In this section, the proposed MFNet is applied to action recognition problems and the experimental results of MFNet are compared with those of other action recognition methods. As datasets, Jester [1] and Something-Something [8] are used because these cannot be easily recognized by just seeing a frame as already mentioned in Section 1. They also are suitable for observing the effectiveness of the proposed motion blocks. We also perform comprehensive ablation studies to prove the effectiveness of the MFNets.

4.1 Experiment Setup

To conduct comprehensive ablation studies on video classification tasks with motion blocks, first we describe our base network framework.

Base Network Framework We select the TSN framework [33] as our base network architecture to train MFNet. TSN is an effective and efficient video processing framework for action recognition tasks. TSN samples a sequence of frames from an entire video and aggregates individual predictions into a video-level score. Thus, TSN framework is well suited for our motion blocks because each block directly extracts the temporal relationships between adjacent snippets in a batch manner.

In this paper, we mainly choose ResNet [10] as our base network to extract spatial feature maps. For the sake of clarity, we divide it into 6 stages. Each stage has a number of stacked residual blocks and each block is composed of several convolutional and batch normalization [11] layers with Rectified Linear Unit (ReLU) [19] for non-linearity. The final stage consists of a global pooling layer and a classifier. Our base network differs from the original ResNet in that it contains the max pooling layer in the first stage. Except this, our base network is the same as the conventional ResNet. The backbone network can be replaced by any other network architecture and our motion blocks can be inserted into the network all in the same way regardless of the type of the network used.

Motion Blocks To form MFNet, we insert our motion blocks into the base network. In case of using ResNet, each motion block is located right after the last residual block of every stage except for the last stage (global pooling and classification layers). Then, MFNet automatically learns to represent spatio-temporal information from consecutive frames, leading the conventional base CNN to extract richer information that combines both appearance and motion features. We also add an 1×1 convolution before each motion block to reduce the number of channels. Throughout the paper, we reduce the number of input channels to motion block by a factor of 16 with the 1×1 convolutional layer. We add a batch normalization layer after the 1×1 convolution to adjust the scale to fit to the features in the backbone network.

Training In the datasets of Jester and Something-Something, RGB images extracted from videos at 12 frames per second with a height of 100 pixels are provided. To augment training samples, we exploit random cropping method with scale-jittering. The width and height of a cropped image are determined by multiplying the shorter side of the image by a scale which is randomly selected in the set of $\{1.0, 0.875, 0.75, 0.625\}$. Then the cropped image is resized to 112×112, because the width of the original images is relatively small compared to that of other datasets. Note that we do not adopt random horizontal flipping to the cropped images of Jester dataset, because some classes are a symmetrical pair, such as *'Swiping Left'* and *'Swiping Right'*, and *'Sliding Two Fingers Left'* and *'Sliding Two Fingers Right'*.

Since motion block extracts temporal motion features from adjacent feature maps, a frame interval between frames is a very important hyper-parameter. We have trained our model with the fixed-time sampling strategy. However, in our experiments, it leads to worse results than the random sampling strategy in

Table 1. Top-1 and Top-5 classification accuracies for different networks with different numbers of training segments $(3, 5, 7)$. The compared networks are TSN baseline, MFNet concatenation version (MFNet-C), and MFNet element-wise sum version (MFNet-S) on Jester and Something-Something validation sets. All models use ResNet-50 as a backbone network and are trained from scratch.

Dataset		Jester		Something-Something	
model	K	top-1 acc.	top-5 acc.	top-1 acc.	top-5 acc.
Baseline	3	82.4%	98.9%	6.6%	21.5%
	5	82.8%	98.9%	9.8%	28.6%
	7	81.0%	98.5%	8.1%	24.7%
MFNet-C50	3	90.4%	99.5%	17.4%	42.6%
	5	95.1%	99.7%	31.5%	61.9%
	7	96.1%	99.7%	37.3%	67.2%
MFNet-S50	3	91.0%	99.6%	15.4%	39.2%
	5	95.6%	99.8%	28.7%	59.1%
	7	96.3%	99.8%	37.1%	67.8%

[33]. With a random interval, the method forces the network to learn through frames composed of various intervals. Interestingly, we get better performance on Jester and Something-Something datasets with the temporal sampling interval diversity.

We use the stochastic gradient descent algorithm to learn network parameters. The batch size is set to 128, the momentum is set to 0.9 and weight decay is set to 0.0005. All MFNets are trained from scratch and we train our models with batch normalization layers [11]. The learning rate is initialized as 0.01 and decreases by a factor of 0.1 for every 50 epochs. The training procedure stops after 120 epochs. To mitigate over-fitting effect, we adopt dropout [25] after the global pooling layer with a dropout ratio of 0.5. To speed up training, we employ a multi-GPU data-parallel strategy with 4 NVIDIA TITAN-X GPUs.

Inference We select equi-distance 10 frames without the random shift. We test our models on sampled frames whose image size is rescaled to 112×112. After that, we aggregate separate predictions from each frame and average them before softmax normalization to get the final prediction.

4.2 Experimental Results

The Jester[1] is a crowd-acted video dataset for generic human hand gestures recognition. It consists $118,562$ videos for training, $14,787$ videos for validation, and $14,743$ videos for testing. The Something-Something[8] is also a crowd-acted densely labeled video dataset of basic human interactions with daily objects. It contains $86,017$ videos for training, $11,522$ videos for validation, and $10,960$ videos for testing. Each of both datasets is for the action classification task involving 27 and 174 human action categories respectively. We report validation

Table 2. Top-1 and Top-5 classification accuracies for different depths of MFNet's base network. ResNet[10] is used as the base network. The values are on JESTER and Something-Something validation sets. All models are trained from scratch, with 10 segments.

Dataset		Jester		Something-Something	
model	backbone	top-1 acc.	top-5 acc.	top-1 acc.	top-5 acc.
MFNet-C	ResNet-18	96.3%	99.8%	39.4%	69.1%
	ResNet-50	96.6%	99.8%	40.3%	70.9%
	ResNet-101	96.7%	99.8%	43.9%	73.1%
	ResNet-152	96.5%	99.8%	43.0%	73.2%

results of our models on the validation sets, and test results from the official leaderboards[5,6].

Evaluation on The Number of Segments Due to the nature of our MFNet, the number of segments, K, in the training is one of the important parameters. Table 1 shows the comparison results of different models while changing the number of segments from 3 to 7 with the same evaluation strategies. We observe that as the number of segments increases, the performance of overall models increases. The performance of the MFNet-C50 (which means that MFNet concatenate version with ResNet-50 as a backbone network) with 7 segments is by far the better than the same network with 3 segments: 96.1% vs. 90.4% and 37.3% vs. 17.4% on Jester and Something-Something datasets respectively. The trend is the same for MFNet-S50, the network with element-wise sum. Also, unlike baseline TSN, MFNets show significant performance improvement as the number of segments increases from 3 to 5.

These improvements imply that increasing K reduces the interval between sampled frames which allows our model to extract richer information. Interestingly, MFNet-S achieves slightly higher top-1 accuracy(0.2% to 0.6%) than MFNet-C on Jester dataset, and MFNet-C shows better performance(0.2% to 2.8%) than MFNet-S on Something-Something dataset. On the other hand, because the TSN baseline is learned from scratch, performance was worse than expected. It can be seen that TSN spatial model without pre-training barely generates any action-related visual features in Something-Something dataset.

Comparisons of Network Depths Table 2 compares the performances as the depths of MFNet's backbone network changes. In the table, we can see that MFNet-C with ResNet-18 achieves comparable performance as the 101-layered ResNet using almost 76% fewer parameters (11.68M vs. 50.23M). It is generally known that as CNNs become deeper, more features can be expressed [10, 23, 27]. However, one can see that because most of the videos in Jester dataset

[5] https://www.twentybn.com/datasets/jester
[6] https://www.twentybn.com/datasets/something-something

Table 3. Comparison of the top-1 and top-5 validation results of various methods on Jester and Something-something datasets. K denotes the number of training segments. The results of other models are from their respective papers.

Dataset	Jester		Something-Something	
model	top-1 acc.	top-5 acc.	top-1 acc.	top-5 acc.
Pre-3D CNN + Avg[8]	-	-	11.5%	30.0%
MultiScale TRN[37]	93.70%	99.59%	33.01%	61.27%
MultiScale TRN (10-crop)[37]	95.31%	99.86%	34.44%	63.20%
MFNet-C50, $K = 7$	96.13%	99.65%	37.31%	67.23%
MFNet-S50, $K = 7$	96.31%	99.80%	37.09%	67.78%
MFNet-C50, $K = 10$	96.56%	99.82%	40.30%	70.93%
MFNet-S50, $K = 10$	96.50%	99.86%	39.83%	70.19%
MFNet-C101, $K = 10$	96.68%	99.84%	43.92%	73.12%

Table 4. Selected test results on the Jester and Something-Something datasets from the official leaderboards. Since the test results are continuously updated, some results that are not reported or whose description is missing are excluded. The complete list of test results is available on official public leaderboards. Our results are based on ResNet-101 with $K = 10$ and trained from scratch. For submissions, we use the same evaluation strategies as the validation mode.

Jester		Something-Something	
model	top-1 acc.	model	top-1 acc.
BesNet (from [37])	94.23%	BesNet (from [37])	31.66%
MultiScale TRN [37]	94.78%	MultiScale TRN [37]	33.60%
MFNet-C101 (ours)	96.22%	MFNet-C101 (ours)	37.48%

are composed of almost similar kinds of human appearances, the static visual entities are very little related to action classes. Therefore, the network depth does not appear to have a significant effect on performance. In Something-Something case, accuracy gets also saturated. It could be explained that generalization of a model seems to be difficult without pre-trained weights on other large-scale datasets, such as Imagenet [4] and Kinetics [14].

Comparisons with The State-of-the-art Table 3 shows the top-1 and top-5 results on the validation set. Our models outperform Pre-3D CNN + Avg [8] and the MultiScale TRN [37]. Because Jester and Something-Something are recently released datasets in the action recognition research field, we also report the test results on the official leaderboards for each dataset for comparison with previous studies. Table 4 shows that MFNet achieves comparable performance to the state-of-the-art methods with 96.22% and 37.48% top-1 accuracies on Jester and Something-Something test datasets respectively on official leaderboards. Note that we do not introduce any other modalities, ensemble methods or pre-trained initialization weights on large-scale datasets such as ImageNet [4] and Kinetics

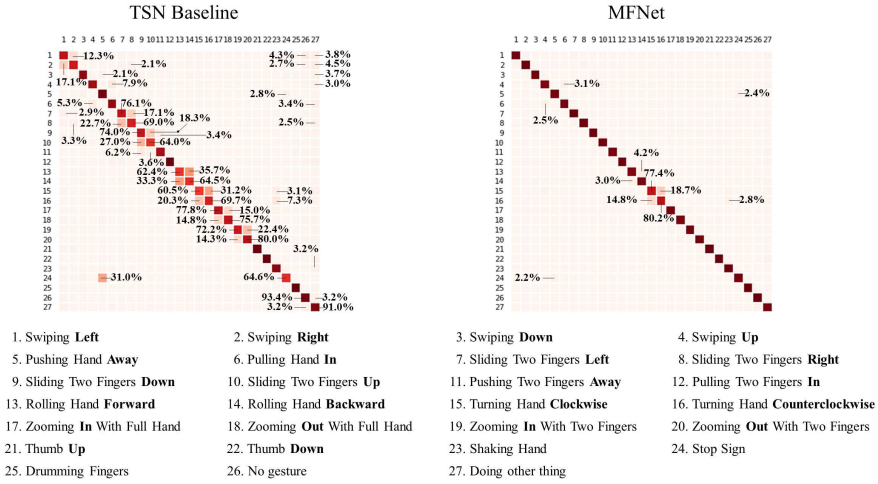


Fig. 5. Confusion matrices of TSN baseline and our proposed MFNet on Jester dataset. The figure is best viewed in an electronic form.

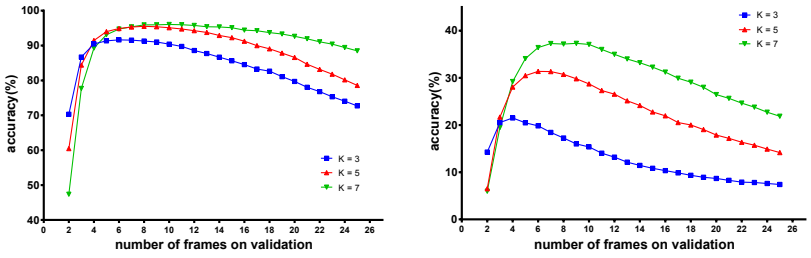
[14]. We only utilize officially provided RGB images as the input of our final results. Also, without 3D ConvNets and additional complex testing strategies, our method provides competitive performances on the Jester and Something-Something datasets.

4.3 Analysis on The Behavior of MFNet

Confusion Matrix We analyze the effectiveness of MFNet comparing with the baseline. Figure 5 shows the confusion matrices of TSN baseline (left) and MFNet (right) on Jester dataset. Class numbers and the corresponding class names are listed below. Figure 5 suggests that the baseline model confuses one action class with its counterpart class. That is, it has trouble classifying temporally symmetric action pairs. For example, (*'Swiping Left', 'Swiping Right'*) and (*'Two Finger Down', 'Two Finger Up'*) are temporally symmetric pairs.

In case of baseline, it predicts an action class by simply averaging the results of sampled frames. Consequently, if there is no optical flow information, it might fail to distinguish some temporal symmetric action pairs. Specifically, we get 62.38% accuracy on *'Rolling Hand Forward'* class among 35.7% of which is misclassified as *'Rolling Hand Backward'*. In contrast, our MFNet showed significant improvement over baseline model as shown in Figure 5 (right). In our experiments, we get the accuracy of 94.62% on *'Rolling Hand Forward'* class among 4.2% of which is identified as *'Rolling Hand Backward'*. It proves the ability of MFNet in capturing the motion representation.

Varying Number of Segments in The Validation Phase We evaluated the models which have different numbers of frames in the inference phase. Figure 6



(a) Jester (b) Something-Something

Fig. 6. Validation accuracies trained with the different number of segments K, while varying the number of validation segments from 2 to 25. The x-axis represents the number of segments at inference time and the y-axis is the validation accuracy of the MFNet-C50 trained with different K.

shows the experimental results of MFNet-C50 on Jester (left) and Something-Something (right) datasets. As discussed in Section 4.2, K, the number of segments in the training phase is a crucial parameter on performance. As we can see, overall performance for all the number of validation segments is superior on large K (7). Meanwhile, the optimal number of validation segments for each K is different. Interestingly, it does not coincide with K but is slightly larger than K. Using more segments reduces the frame interval which allows extracting more precise spatio-temporal features. It brings the effect of improving performance. However, it does not last if the numbers in the training and the validation phases differ much.

5 Conclusions

In this paper, we present MFNet, a unified network containing appearance blocks and motion blocks which can represent both spatial and temporal information for action recognition problems. Especially, we propose the motion filter that outputs the motion features by performing the shift operation with the fixed set of predefined directional filters and subtracting the resultant feature maps from the feature maps of the preceding frame. This module can be attached to any existing CNN-based network with a small additional cost. We evaluate our model on two datasets, Jester and Something-Something, and obtain outperforming results compared to the existing results by training the network from scratch in an end-to-end manner. Also, we perform comprehensive ablation studies and analysis on the behavior of MFNet to show the effectiveness of our method. In the future, we will validate our network on large-scale action recognition dataset and additionally investigate the usefulness of the proposed motion block.

References

1. The 20bn-jester dataset. https://www.twentybn.com/datasets/jester
2. Carreira, J., Zisserman, A.: Quo vadis, action recognition? a new model and the kinetics dataset. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4724–4733. IEEE (2017)
3. Dalal, N., Triggs, B., Schmid, C.: Human detection using oriented histograms of flow and appearance. In: European conference on computer vision. pp. 428–441. Springer (2006)
4. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on. pp. 248–255. IEEE (2009)
5. Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., Van Der Smagt, P., Cremers, D., Brox, T.: Flownet: Learning optical flow with convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2758–2766 (2015)
6. Feichtenhofer, C., Pinz, A., Wildes, R.: Spatiotemporal residual networks for video action recognition. In: Advances in neural information processing systems. pp. 3468–3476 (2016)
7. Feichtenhofer, C., Pinz, A., Zisserman, A.: Convolutional two-stream network fusion for video action recognition (2016)
8. Goyal, R., Kahou, S.E., Michalski, V., Materzynska, J., Westphal, S., Kim, H., Haenel, V., Fruend, I., Yianilos, P., Mueller-Freitag, M., et al.: The something something video database for learning and evaluating visual common sense. In: Proc. ICCV (2017)
9. Hara, K., Kataoka, H., Satoh, Y.: Learning spatio-temporal features with 3d residual networks for action recognition. In: Proceedings of the ICCV Workshop on Action, Gesture, and Emotion Recognition. vol. 2, p. 4 (2017)
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
11. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International conference on machine learning. pp. 448–456 (2015)
12. Jiang, Y., Liu, J., Zamir, A.R., Toderici, G., Laptev, I., Shah, M., Sukthankar, R.: Thumos challenge: Action recognition with a large number of classes (2014)
13. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Fei-Fei, L.: Large-scale video classification with convolutional neural networks. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 1725–1732 (2014)
14. Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., et al.: The kinetics human action video dataset. arXiv preprint arXiv:1705.06950 (2017)
15. Kuehne, H., Jhuang, H., Garrote, E., Poggio, T., Serre, T.: HMDB: a large video database for human motion recognition. In: Proceedings of the International Conference on Computer Vision (ICCV) (2011)
16. Kuehne, H., Jhuang, H., Stiefelhagen, R., Serre, T.: Hmdb51: A large video database for human motion recognition. In: High Performance Computing in Science and Engineering 12, pp. 571–582. Springer (2013)

17. LeCun, Y., Bengio, Y., et al.: Convolutional networks for images, speech, and time series. The handbook of brain theory and neural networks **3361**(10), 1995 (1995)
18. Miech, A., Laptev, I., Sivic, J.: Learnable pooling with context gating for video classification. arXiv preprint arXiv:1706.06905 (2017)
19. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th international conference on machine learning (ICML-10). pp. 807–814 (2010)
20. Ng, J.Y.H., Choi, J., Neumann, J., Davis, L.S.: Actionflownet: Learning motion representation for action recognition. arXiv preprint arXiv:1612.03052 (2016)
21. Ng, J.Y.H., Hausknecht, M., Vijayanarasimhan, S., Vinyals, O., Monga, R., Toderici, G.: Beyond short snippets: Deep networks for video classification. In: Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on. pp. 4694–4702. IEEE (2015)
22. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In: Advances in neural information processing systems. pp. 568–576 (2014)
23. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
24. Soomro, K., Zamir, A.R., Shah, M.: Ucf101: A dataset of 101 human actions classes from videos in the wild. arXiv preprint arXiv:1212.0402 (2012)
25. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research **15**(1), 1929–1958 (2014)
26. Sun, S., Kuang, Z., Ouyang, W., Sheng, L., Zhang, W.: Optical flow guided feature: A fast and robust motion representation for video action recognition. CoRR **abs/1711.11152** (2017), http://arxiv.org/abs/1711.11152
27. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A., et al.: Going deeper with convolutions. Cvpr (2015)
28. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: Computer Vision (ICCV), 2015 IEEE International Conference on. pp. 4489–4497. IEEE (2015)
29. Tran, D., Ray, J., Shou, Z., Chang, S.F., Paluri, M.: Convnet architecture search for spatiotemporal feature learning. arXiv preprint arXiv:1708.05038 (2017)
30. Wang, H., Kläser, A., Schmid, C., Liu, C.L.: Action recognition by dense trajectories. In: Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on. pp. 3169–3176. IEEE (2011)
31. Wang, H., Schmid, C.: Action recognition with improved trajectories. In: Computer Vision (ICCV), 2013 IEEE International Conference on. pp. 3551–3558. IEEE (2013)
32. Wang, L., Li, W., Li, W., Van Gool, L.: Appearance-and-relation networks for video classification. arXiv preprint arXiv:1711.09125 (2017)
33. Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Van Gool, L.: Temporal segment networks: Towards good practices for deep action recognition. In: European Conference on Computer Vision. pp. 20–36. Springer (2016)
34. Wu, B., Wan, A., Yue, X., Jin, P., Zhao, S., Golmant, N., Gholaminejad, A., Gonzalez, J., Keutzer, K.: Shift: A zero flop, zero parameter alternative to spatial convolutions. arXiv preprint arXiv:1711.08141 (2017)
35. Wu, Z., Jiang, Y.G., Wang, X., Ye, H., Xue, X., Wang, J.: Fusing multi-stream deep networks for video classification. arXiv preprint arXiv:1509.06086 (2015)

36. Zhang, B., Wang, L., Wang, Z., Qiao, Y., Wang, H.: Real-time action recognition with enhanced motion vector cnns. In: Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on. pp. 2718–2726. IEEE (2016)
37. Zhou, B., Andonian, A., Torralba, A.: Temporal relational reasoning in videos. arXiv preprint arXiv:1711.08496 (2017)
38. Zhu, Y., Lan, Z., Newsam, S., Hauptmann, A.G.: Hidden two-stream convolutional networks for action recognition. arXiv preprint arXiv:1704.00389 (2017)