Real-Time Action Recognition With Deeply Transferred Motion Vector CNNs

Bowen Zhang, Limin Wang, Zhe Wang, Yu Qiao, *Senior Member, IEEE*, and Hanli Wang, *Senior Member, IEEE*

***Abstract*— The two-stream CNNs prove very successful for video-based action recognition. However, the classical two-stream CNNs are time costly, mainly due to the bottleneck of calculating optical flows (OFs). In this paper, we propose a two-stream-based real-time action recognition approach by using motion vector (MV) to replace OF. MVs are encoded in video stream and can be extracted directly without extra calculation. However, directly training CNN with MVs degrades accuracy severely due to the noise and the lack of fine details in MVs. In order to relieve this problem, we propose four training strategies which leverage the knowledge learned from OF CNN to enhance the accuracy of MV CNN. Our insight is that MV and OF share inherent similar structures which allow us to transfer knowledge from one domain to another. To fully utilize the knowledge learned in OF domain, we develop deeply transferred MV CNN. Experimental results on various datasets show the effectiveness of our training strategies. Our approach is significantly faster than OF based approaches and achieves processing speed of 390.7 frames per second, surpassing real-time requirement. We release our model and code to facilitate further research.**[1]

***Index Terms*— Action recognition, motion vector, knowledge transfer, real-time processing, deep learning.**

Manuscript received May 31, 2017; revised October 18, 2017 and December 8, 2017; accepted December 24, 2017. Date of publication January 8, 2018; date of current version February 21, 2018. This work was supported in part by the National Natural Science Foundation of China under Grant U1613211, Grant 61633021, and Grant 61622115, in part by the Shenzhen Basic Research Program under Grant JCYJ20150925163005055, in part by the External Cooperation Program of BIC Chinese Academy of Sciences under Grant 172644KYSB20150019 and Grant 172644KYSB20160033, and in part by the Shanghai Engineering Research Center of Industrial Vision Perception and Intelligent Computing under Grant 17DZ2251600. This work was mainly conducted when B. Zhang interned in Shenzhen Institutes of Advanced Technology. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Jianfei Cai. *(Corresponding author: Yu Qiao.)*

B. Zhang is with The Key Laboratory of Embedded System and Service Computing, Department of Computer Science and Technology, Ministry of Education, Tongji University, Shanghai 200092, China (e-mail: zbwglory@gmail.com).

L. Wang is with the Computer Vision Laboratory, ETH Zurich, 8092 Zurich, Switzerland (e-mail: 07wanglimin@gmail.com).

Z. Wang is with the Computational Vision Group, University of California at Irvine, Irvine, CA 92697 USA (e-mail: buptwangzhe2012@gmail.com).

Y. Qiao is with the Guangdong Provincial Key Laboratory of Computer Vision and Virtual Reality Technology, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China (e-mail: yu.qiao@siat.ac.cn).

H. Wang is with the Key Laboratory of Embedded System and Service Computing, Department of Computer Science and Technology, Ministry of Education, Tongji University, Shanghai 200092, China (e-mail: hanliwang@tongji.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2018.2791180

[1] https://github.com/zbwglory/MV-release

I. Introduction

HUMAN action recognition [1]–[5] has been extensively studied in the recent years, due to its great potential in real applications like video surveillance, video retrieval, and human computer interaction. The aim of action recognition is to automatically classify actions in real world videos efficiently and effectively. In recent years, many methods have been proposed to boost the classification accuracy. These algorithms are mainly based on two paradigms: Bag-of-Visual-Words framework [6] and deep learning [1]. Bag-of-Visual-Words framework (or its variants) is a stage-wise method, whose pipeline contains feature extraction, feature encoding, and classification. One popular method along this line is the improved-dense-trajectories (iDT) [2] and Fisher Vector encoding [7]. It uses trajectory-aligned hand-crafted features to represent action and achieves high accuracy on various datasets. Unlike these traditional methods, deep learning is an end-to-end framework. It takes a video as input and employs multiple-layer neural network as its architecture. The parameters of deep learning method are automatically tuned based on back propagation algorithm. Two-stream CNN is a successful architecture in this deep learning paradigm. Both RGB CNN and optical flow CNN are used to extract appearance and motion representation from videos, respectively. These representations are used to predict action classes from videos. Two-stream framework and its variants [8] achieve the state-of-the-art accuracy on several large datasets like UCF101 [9] and HMDB51 [10]. The existing two-stream frameworks mainly rely on optical flow extraction to represent motion information. However the calculation of optical flow is time consuming, which prohibits the real-time preprocessing of two-stream based approaches even with GPU.

The main objective of this paper is to develop a real-time action recognition approach with high performance and accuracy. We utilize the successful two-stream framework [1] as our basic architecture. It is non-trivial to speed up performance while still keep the recognition accuracy of two-stream CNNs, because it requires optical flow as its input. Only performing prediction with RGB image leads to inferior recognition accuracy. However the calculation of optical flow is computationally expensive, for example it can only be conducted at the speed of 16.7 frames per second (fps) with K40 GPU [11], which is a bottleneck for real-time processing. To circumvent this problem, in this paper, motion vector as the input of CNN is introduced. Motion vector is encoded

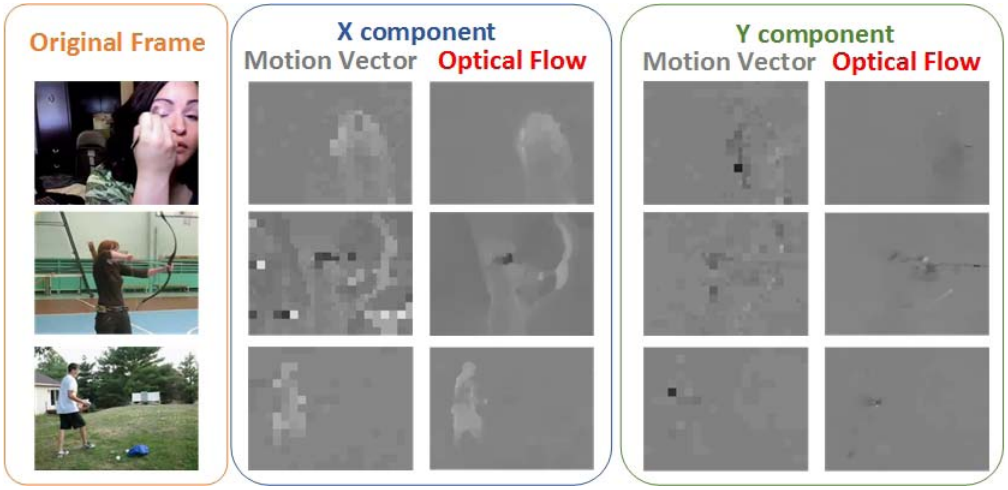


Fig. 1. Comparison of motion vector and optical flow in x and y components. We can see that motion vector contains lots of noisy movement information and it is much coarser than optical flow. It is clearly that the structures of bow and arrow are lost and the outline of human is blurred.

in the video stream during video compression phrase and can be directly extracted almost with no extra computational overhead.

Motion vector [12] is originally proposed for video coding. It is designed to exploit the motion information of corresponding image blocks to reduce the bit rate of video. Reference [13] shows that motion vector can be used for action recognition. Similar to optical flow, motion vector contains local motion information. However, as it is not designed to reveal the motion as accurate as possible, motion vector contains noisy and imprecise movement information as shown in Fig. 1. Directly using motion vector as input can degrade the accuracy severely.

To solve the possible issue of accuracy drop, our key idea in this paper is to transfer the knowledge from optical flow CNN to motion vector CNN to improve its generalization ability. Compared with optical flow images, motion vector images contain similar movement information. The main difference lies in the quality of motion. Motion vector contains *coarse* and *noisy* movement pattern, while optical flow contains more precise and clear one. Due to the high quality of optical flow images, optical flow CNN can learn elaborate and concise filters, while motion vector CNN only learn noisy filters which harms its recognition accuracy. This fact inspires us that the knowledge learned by optical flow CNN may be beneficial to motion vector CNN. To fully unleash the potential of motion vector CNN, we design an algorithm, called deeply-connected transfer, to perform multi-layer knowledge transfer from optical flow CNN to motion vector CNN. It should be noted that optical flows are only used during training phrase for knowledge transfer. For testing, only motion vectors are used for real time action recognition. Experiments show that deeply-transferred motion vector CNN obtains a significant accuracy improvement over directly using motion vector as input, while still keeps the good merit for real-time processing speed of motion vectors.

The preliminary version is published in CVPR 2016 [14] and we have extended it in two important ways. First, we propose a new training strategy to improve motion vector CNN's accuracy. Second, more extensive experiments are performed to verify the effectiveness of our new approach. In particular, we observe that the learning strategies proposed in [14] are kinds of shallow supervision. During training, optical flow CNN's knowledge is only presented to the final layer of motion vector net. The knowledge that contained in middle layers of optical flow CNN is not directly exploited. In this paper, a new method called "Deeply Connected Transfer" is proposed to enable knowledge transfer between middle layers. Furthermore, extensive experiments are performed on the UCF101 [9], HMDB51 [10], and THUMOS14 [15] datasets. These results verify that our newly proposed method can provide stronger supervision than [14] and further improve the accuracy of motion vector CNN.

II. Related Work

Recent years have witnessed significant progress for action recognition. State-of-the-art methods can be roughly divided into two frameworks: the Bag-of-Visual-Words (BoVW) paradigm and deep learning approach.

BoVW framework is originally proposed for image classification [16]. Further researches show its potential for video retrieval [6] and action recognition [2]. Typical BoVW framework consists of three steps: feature extraction, feature encoding, and classification. For feature extraction, Laptev [17] explored the spatial-temporal domain by extending harris corner detector to 3D. Wang *et al.* [18] evaluated different combinations of detectors and descriptors, and showed that dense feature extraction can exhibit better accuracy than salient point based approach. Wang *et al.* [19] further extended this idea by utilizing optical flow to track dense feature points in several continuous frames and employed several descriptors to extract feature along optical flow trajectories. Wang and Schmid [2] discovered that camera motion can

hamper accuracy for action recognition and exploited camera motion elimination method to further improve the accuracy. Popular descriptors for action recognition include HOG [20], HOF [17], MBH [21], and the recently proposed TDD [5]. Unlike the previous hand-crafted features [17], [20], [21], TDD utilized deep neural network to extract features along trajectories and showed significant improvement on various datasets. For descriptor encoding, hard quantization [6], VLAD [22] and Fisher Vector [7] are among the popular methods for action recognition. Peng *et al.* [23] conducted extensive experiments on various datasets to give practical advice on how to choose the optimal setup of feature encoding for action recognition.

Recent studies demonstrated that deep learning approaches can achieve superior accuracy on image classification [24] and object detection [25], which inspires researchers to utilize CNN for action recognition task. Different from image classification, video based action recognition is in spatial-temporal domain, where motion information yields an important cue. One research line is to exploit contiguous frame relationship by stacking RGB images. Karpathy *et al.* [26] first used CNN on stacked RGB images to learn motion patterns, and designed several temporal pooling method. Tran *et al.* [4] proposed to use 3D convolution to directly extract motion relation in stacked RGB images, and showed good speed and accuracy on various datasets. Wang *et al.* [27] proposed a new module, called as *SMART*, to directly model appearance and relation from RGB images in an explicit and separate way. Another research line is based on optical flow. Optical flow can directly unveil motion information by calculating the movement of corresponding points. One successful approach in this line is two-stream CNNs [1], where a two-stream net was developed to exploit appearance information and motion relation in RGB CNN and optical flow CNN, respectively. Two-stream CNN framework is used as a baseline for our study as it achieves high recognition accuracy. Wang *et al.* [8] improved two stream framework by using temporal segments of video to train multiple snippet-level CNNs, which achieves state-of-the-art accuracy on several datasets. Feichtenhofer *et al.* [28] explored the fusion strategies of RGB CNN and optical flow CNN, which showed high accuracy on action recognition. Feichtenhofer *et al.* [29] proposed ST-ResNet by combining ResNet [25] with two-stream convnets. ST-ResNet achieved impressive results on UCF101 and HMDB51 datasets. Ng *et al.* [30] introduced recurrent neural network (LSTM) to further exploit motion patterns in optical flow images. Wu *et al.* [31] combined the merits of two-stream CNNs and LSTMs by fusing RGB net and optical flow net with a recurrent architecture.

Despite of improving accuracy, several approaches were proposed to accelerate the processing speed of deep neural networks. Knowledge distillation method [32] was proposed to compress cumbersome net. Courbariaux and Bengio [33] discovered that float computation in neural network is time consuming and binary number can be efficiently calculated by shift operations. Courbariaux and Bengio [33] proposed to binarize the activation of feed forward processing to accelerate neural network computation. Rastegari *et al.* [34] further extended this idea by proposing XNOR-Net to achieve high accuracy on ImageNet datasets.

Several researches also aimed to improve processing speed of traditional methods on action recognition. For example, Kantorov and Laptev [13] accelerated dense trajectory method [19] by employing motion vector to replace optical flow. Kantorov and Laptev [13] further used FLANN to substitute brute force search in Fisher Vector [7] and VLAD [22] to improves the speed.

Another thread of researches focused on leveraging privileged knowledge provided by teacher model to improve student model's performance. Vapnik and Izmailov [35] provided detailed investigations on two topics in learning using privileged information (LUPI): similarity control in LUPI paradigm and transfer knowledge from privileged knowledge space to decision space. They proposed SVM+ to implement LUPI algorithm. Lapin *et al.* [36] showed the close relationship between weighted SVM and privileged information. Further, Lopez-Paz *et al.* [37] unified two style of knowledge transfer, privileged information and knowledge distillation, into generalized distillation. They showed that distillation and privileged information can improve the results with respect to pure supervised learning. Li *et al.* [38] proposed sparse multi-instance learning using privileged information (sMIL-PI) approach. sMIL-PI leverages textual features from tags and captions of web images as privileged knowledge to tackle noisy labeling problem in training data. Their work showed that using privileged information can improve the performance of image retrieval and image categorization tasks. Niu *et al.* [39] showed that sMIL-PI can also achieve promising results for action recognition. The idea proposed in our work that using teacher network to improve student net's performance is similar to LUPI paradigm. However, unlike LUPI which is based on SVM framework, our approach implements knowledge transfer in deep learning domain.

Among these approaches, the most relevant works to us are [13], [40], and [32]. Unlike [13], we use motion vector to improve speed in the deep learning framework. It is non-trivial to train a high accuracy network with motion vector. Our work is also related to knowledge distillation [32] in spirit. Unlike [32], however, our aim is not to transfer knowledge in the same domain between two different structures but to transfer knowledge between two different domains with the same structure. We further design several strategies to enhance knowledge transfer from optical flow domain to motion vectors. Unlike [40] that only using the output of one layer as supervision signal, we fully utilize the multiple middle representations to provide deeply knowledge transfer. The soft labels of teacher networks have been also employed in other computer vision tasks for different objectives, such as class disambiguation in large-scale scene recognition [41] and knowledge transfer in CNN fine tuning for event recognition [42].

III. Motion Vector for Deep Action Recognition

Two-stream CNNs [1] consists of two parts, RGB CNN and optical flow CNN, to achieve state-of-the-art accuracy on

various datasets. RGB CNN can be conducted in short time with GPU. However, the optical flow part is computational expensive and cannot satisfy real-time preprocessing requirement. Optical flow CNNs take frames of optical flow as input. It first needs to extract optical flow images from video. Then these images are processed with a CNN. Although the feed forward process of CNNs can be conducted at fast speed (around 300ms for 250 frames with center crop) with GPU, the extraction of optical flow is relatively slow. For example, with Farneback's method [43], calculating optical flow needs 360ms per frame on CPU with efficient implementation. Even with GPU, optical flow (Brox's flow [11]) can only be extracted around 60ms per frame, which is still far from the requirement of real-time processing. Thus, the calculation of optical flow is one of the main bottlenecks that lower the processing speed of two-stream CNNs.

As the optical flow CNN is an important part and has a large contribution to the accuracy of two-stream CNNs. Directly processing video with only RGB CNN degrades the recognition accuracy severely. Here, we follow the two-stream architecture and propose motion vector CNNs to extract the motion pattern of videos, instead of using the computationally expensive optical flows.

Motion vector is similar to optical flow. Both are two-dimension vectors to describe the movement information of corresponding pixels in two continuous frames. Unlike optical flow, motion vector is widely used in various video coding standards (H.264 [12], HEVC [44] and etc.). It is available in compressed video stream and can be obtained directly with almost no computational cost. This property makes motion vector an attractive substitution for optical flow to achieve efficient action analysis. Early work [13] had demonstrated the usefulness of motion vector for action recognition. They purposed to use motion vector to form trajectories and then used VLAD and Fisher vector with efficient implementation to describe videos for action classification. Different to this work, we explore motion vector in deep neural network framework. The main difficulty comes from the noise and the block-wise imprecise motion information exhibited by motion vector images. As shown in our experiments, directly training CNN with motion vectors from scratch will largely harm the recognition accuracy.

To tackle this problem, several training methods are proposed to transfer the knowledge learned by optical flow CNN to motion vector CNN. Here, our insight is that both motion vector and optical flow contain the movement information. Furthermore, the knowledge learned by optical flow CNN and motion vector CNN are correlated. Since optical flow is more precise and optical flow CNN can learn more elaborate filters, we may leverage optical flow CNN as a teacher net to improve the accuracy of motion vector CNN.

A. Motion Vector

In this subsection, we give a brief description of motion vector and explain why it is hard to train motion vector CNN with high accuracy.

Motion vector is originally designed for video coding. In video coding, the main goal is to reduce the spatial and temporal redundancy within several continuous frames. Motion vectors describe the movement information of corresponding blocks in two frames, which yields an ideal cue to exploit the temporal redundancy in two frames. Thus motion vector is widely implemented in various video coding standards like H.264, MPEG, HEVC, and etc. For action recognition, as motion vector is already calculated in video coding phase and contains motion, it can provide effective motion information for action recognition with high efficiency.

However, training motion vector CNN with high accuracy is challenging. Firstly, motion vector only provides block wise movement information. In video coding, macro block is the basic coding unit, which has size ranging from 8×8 to 16×16. Different with the pixel level motion information provided by optical flow, motion vector only yields macro block level information. This property causes the coarse structure of motion vector, as Fig. 1. It also poses difficulty for using motion vector CNN to learn fine-grained action information, which further degrades the accuracy of motion vector CNN severely.

Furthermore, motion vectors contain noisy motion information, which made it difficult to learn high accuracy motion vector CNN. As stated before, motion vector needs to meet the balance between the speed of encoding and the bit rate of video. It is calculated based on three or four steps of block-wise comparison. Thus, motion vectors fail to provide precise motion information. It can be clearly seen from Fig. 1, unlike the clear background part of optical flow image, noise patterns exist in motion vector image. This property hampers motion vector CNN to achieve high accuracy.

Motion vectors may not exist in all frames. It is calculated based on reference frames. In order to achieve the balance of video quality and the compression rate, images in video coding are grouped into group-of-pictures (GOP). One typical GOP contains intra-coded frame (I-frame), predictive frame (P-frame), and bi-predictive frame (B-frame). As the name indicated, I-frame is coded based on itself which indicates that it contains no motion vector. P-frame and B-frame are coded based on other reference frames, which means that they both contain movement information. Empty I-frame poses difficulties in training high accuracy CNN. Our strategy to this is to use previous frame's motion vector to replace empty I-frame. We find this simple strategy works well in practice.

B. Real-Time Action Recognition Frameworks

The proposed real-time action recognition framework is shown in Fig. 2. It contains two components: video decoder and two-stream architecture CNNs. The video decoder takes compressed video as input and directly gets RGB and motion vector images during decoding phrase. Then RGB and motion vector images are fed into two-stream CNNs to get the action prediction of this video. The main difference between our proposed real-time action recognition framework with two-stream CNNs is that our method doesn't require optical flow computation during deploying. Motion vector CNN (MV-CNN) and RGB CNN are utilized in our framework to extract high level motion and appearance representations, respectively. Thus, the most time-consuming part, optical flow calculation, is avoided in our real-time action recognition framework.

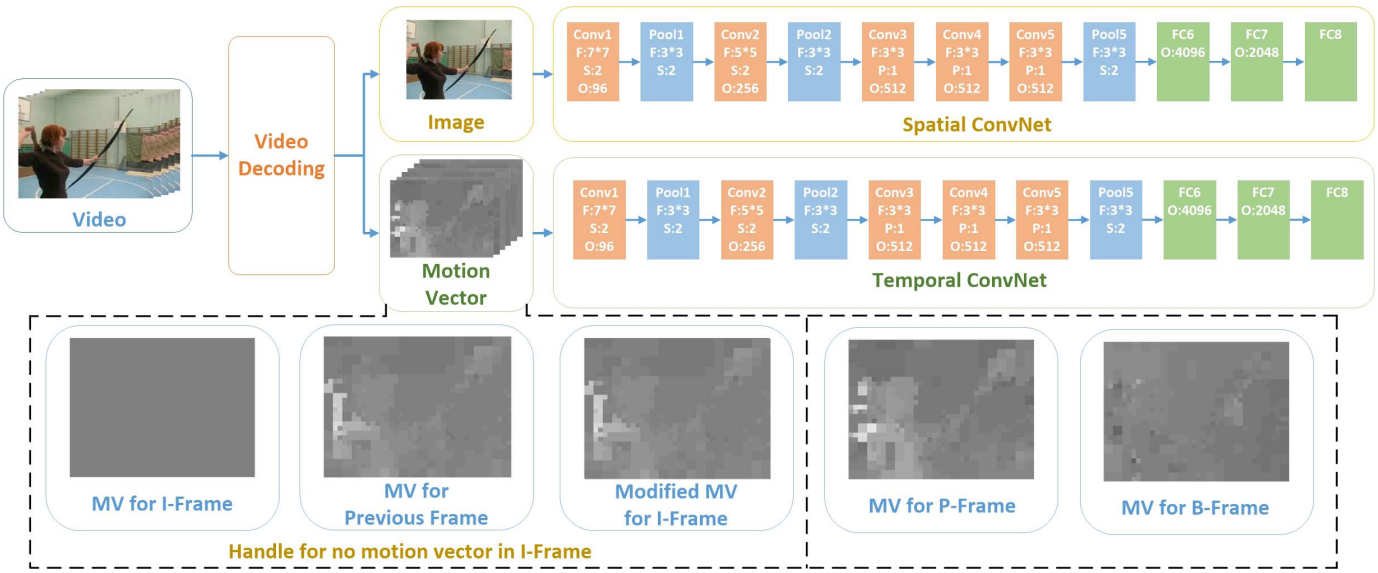


Fig. 2. Structure for real-time action recognition system. In spatial and temporal CNN, F stands for kernel size and S means stride step. O represents for output number and P is pad size. Each time, MV-CNN processes an input by stacking 10 motion vectors, which contains 20 channels in total (10 channels for x axis and 10 channels for y axis). RGB-CNN processes one RGB image with 3 channels at one time.

In training phase, RGB images and motion vector are firstly extracted from video. The video's label is assigned to each frame. Data augmentation is important for training CNN. Random cropping and random scale jitter in spatial domain are employed to get a patch from image and motion vector.

In testing phases, videos are firstly decomposed into raw images and motion vectors. RGB CNN and MV-CNN are then employed to take images and motion vectors as input. The final action prediction of video is determined by the weighted average of two CNN's prediction scores. Weights are set as 1 and 2 for spatial CNN and temporal CNN respectively.

For fair comparison with two stream CNNs [1], ClarifaiNet [1] is used as the basic architecture for spatial and temporal CNN. Following [1] and [45], dropout ratios are set to 0.5 for spatial net to avoid over-fitting. For temporal net, dropout ratios are set to 0.9 and 0.8 for FC6 and FC7 respectively. Our spatial net is pre-trained on ImageNet ILSVRC-2012 dataset and then fine-tuned on action dataset. The learning rate for spatial net is firstly set to 10^{-3} and then drop to 10^{-4} after 14k iterations. The whole training process terminates at 20k steps. As HMDB51 dataset is relatively smaller than UCF101 and THUMOS14 dataset is larger than UCF101, we stops the training process at 10k steps and 50k steps, respectively.

For temporal net, we slightly modify the ReLU layers of original ClarifaiNet to PReLU layers, as it leads to better results and accelerates convergence. 10 frames of motion vectors is stacked as input for temporal net. Learning rate for temporal CNN starts from 10^{-2} and then decreases to 10^{-3} after 30k iterations. Learning rate further drops to 10^{-4} at 70k steps. The training stops at 90k iterations.

IV. Deeply-Transferred Motion Vector CNNs

As motion vectors only contain block level details, and suffer from noisy and imprecise motion information. It is challenging to train motion vector CNN with high accuracy. Experiments show that directly using motion vector to replace optical flow will lead to 7%, 10% and 26% accuracy degradation on UCF101 split1, HMDB split1 and THUMOS14 datasets, respectively. Our aim is to achieve the real-time processing merit of motion vector as well the high recognition accuracy as optical flow. Although motion vector and optical flow are calculated in different domains, they both contain similar motion information. Inspired by this fact, we design several methods to leverage the rich and fine-grained features that learned by optical flow CNN to improve motion vector CNN. These methods can be seen as transfer the knowledge learned in optical domain flow to that in motion vector domain. For training, as in Fig. 3(b), optical flow CNN is employed as teacher network to transfer knowledge to the student network: motion vector CNN. For the testing phrase, only motion vector CNN is used as temporal net to process the video. Optical flow images and optical flow CNN need not be calculated in testing. Thus, in testing, our proposed knowledge transfer strategy will not influence the testing speed of our action recognition system.

To implement the idea described above, four different strategies are proposed to transfer knowledge from OF-CNN to MV-CNN. We first introduce several notations. For teacher net in optical flow domain, the parameters are defined as $T_p = \{T_p^1, T_p^2, ..., T_p^n\}$, where T_p^n stands for parameters of the n-th layer of teacher network and n represents the total number of layers. Parameters for student net in motion vector domain are denoted by $S_p = \{S_p^1, S_p^2, ..., S_p^n\}$. For simplicity, we assume that the motion vector CNN has the same structure as optical flow CNN. Our method can also be extended to other structures.

A. Teacher Initialization

Extensive works [1], [8], [46] in image and action classification show that initializing network with a pre-trained

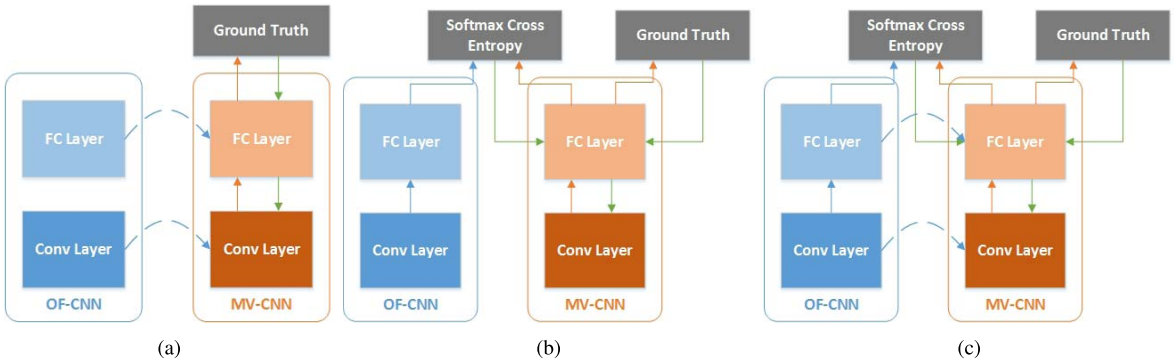


Fig. 3. Structure of Teacher Initialization, Supervision Transfer, and their combination. Blue dash lines represent copying the initial weights from teacher net to student net. Green lines are the backward propagation path. Blue full lines mean feed forward paths of teacher flow. Orange lines are feed forward paths of student net. (a) Strategy A: teacher initialization. (b) Strategy B: Supervision transfer. (c) Strategy C: Combination.

model on ImageNet can improve the accuracy and accelerate convergence. This fact inspires us to find an appropriate pre-trained model for motion vector CNN. Although motion vector and optical flow are from different domains, they are inherently correlated as they both contain similar motion information of local patches. Thus, optical flow CNN is used as a pre-trained model for motion vector CNNs. More specifically, shown in Fig. 3(a), we use optical flow CNN (OF-CNN) to initialize the parameters of motion vector CNN (MV-CNN),

$$S_p^t = T_p^t, \quad t = 1, \ldots, n. \tag{1}$$

Then, motion vector images are used to fine-tune motion vector CNN until convergence. Teacher initialization strategy directly provides MV-CNN a good start point for training with knowledge of detailed motion information. It can be seen as MV-CNN starts to train on fine optical flow features and then learns by itself.

For implementation details, we first set the learning rate as 10^{-3}, and then decrease it to 10^{-4} and 10^{-5} at 30k and 70k steps respectively. The training stops at 90k iterations.

B. Supervision Transfer

Teacher initialization provides student network a good starting point for training with pre-learned network parameters. However, teacher network is not involved in training process. Student network MV-CNN is only trained with motion vector samples. As the inaccurate and coarse nature of motion vectors, the fine motion features provided by the teacher initialization may be diminished during the fine-tuning process. To tackle this problem, as in Fig. 3 (b), we propose to include additional supervision signal by using both ground truth label and teacher network to teach student net during training. Thus, MV-CNN can learn from OF-CNN during the whole training process. Here the last full connection layer of OF-CNN is employed as a new supervision signal for MV-CNN.

The technique in supervision transfer is similar to Hinton *et al.*'s [32] work on dark knowledge. However our aims are different. Hinton's work mainly focuses on how to compress a large network to a small one with similar accuracy. Inputs of two networks are in the same domain. But the structures for cumbersome network and small network are different. In our problem, the input for teacher and student net are different (optical flow vs. motion vector). The main barrier needs to be tackled is how to improve the accuracy of student net with low quality input. Thus, we do not need to compress a network. Our aim is to use teacher network to improve the accuracy of the student with the same structure.

For a given frame I, the optical flow and motion vector are defined as o and v respectively. The output of the last fully connected (FC) layer of teacher CNN and student CNN are calculated as: $T^n(o) = \text{softmax}(T^{n-1}(o))$, and $S^n(v) = \text{softmax}(S^{n-1}(v))$, respectively, where 'softmax' function is employed to generate a probability score of multiple classes from the FC feature.

For transferring knowledge from OF-CNN to MV-CNN, the difference between student's and teacher's output need to be measured. Inspired by Hinton *et al.*'s [32], a *teacher supervision loss* function is introduced. We utilize a temperature parameter *Temp* to soften both teacher's and student's output to ease the learning difficulty. The softmax output of teacher net is softened as $P_T = \text{softmax}(T^{n-1}/Temp)$, Similarly, the student net's softmax output is defined as $P_S = \text{softmax}(S^{n-1}/Temp)$, we use cross-entropy function to define the *teacher supervision loss* (TSL):

$$L_{TSL} = -\sum_{i=1}^{k} P_T(i) \log P_S(i), \tag{2}$$

where k is the dimension of student and teacher's output (number of categories).

In spite of teacher supervision loss, the cross entropy between student's output S^n and the ground truth Q is still needed to be minimized. Thus, the ground truth loss (GT) is defined by,

$$L_{GT} = -\sum_{i} 1[Q = i] \log S^n(i), \tag{3}$$

where S^n and Q represent the unsoftened student net's softmax vectors and the ground truth label respectively.

TSL loss (Eq.2) and GT loss (Eq.3) are combined to form the final loss:

$$L = L_{TSL} + w \cdot L_{GT} \tag{4}$$

where w is a weight to balance these two terms. As suggested in [32], during training, the weight w for TSL and GT loss

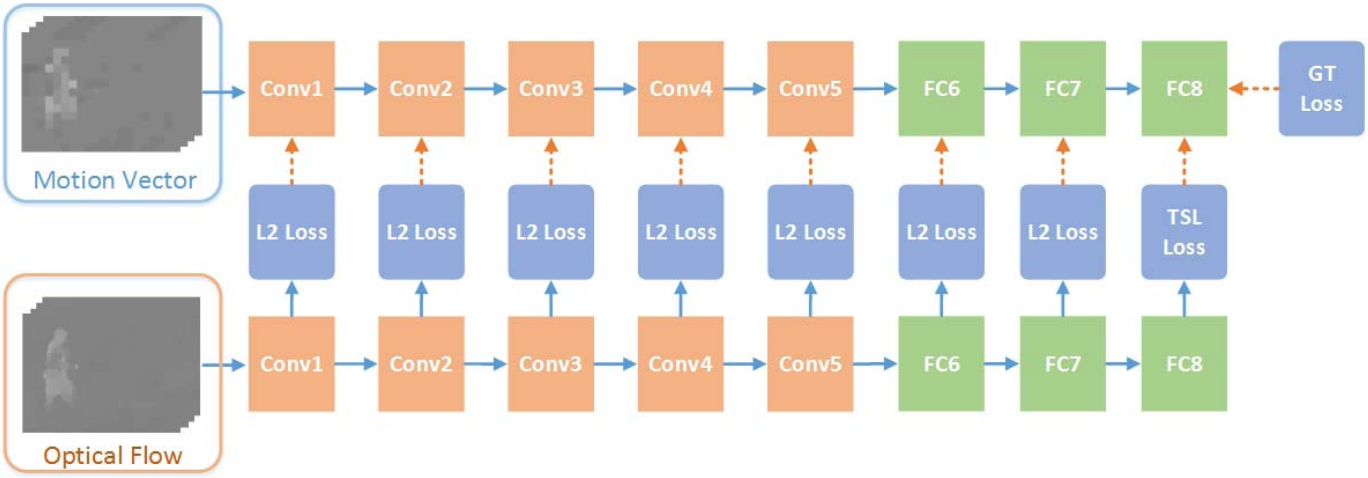


Fig. 4. Structure for Deeply Connected Transfer. Blue lines represent the feed forward process of CNN, while the orange dash line means the back propagation for DTMV-CNN. It should be noticed that the OF-CNN is only utilized during training and the weight for OF-CNN is frozen.

is set as $Temp^2$ to balance gradients of these two losses. The parameters of teacher network is frozen. Only parameters of student net is updated by the supervision of teacher net and ground truth label.

For implementation details, learning rate starts from 10^{-3} and then decays to 10^{-4} at 50k and 10^{-5} at 70k steps. The whole training procedure terminates at 90k iterations.

C. Combination

In third strategy, we want to combine the merits of both teacher initialization and supervision transfer. For the combination strategy, the teacher's parameters are firstly copied to student's net. Then both teacher supervision loss Eq.2 and ground truth loss Eq.3 are employed to transfer the knowledge from OF-CNN to MV-CNN. In this way, the pre-trained model on optical flow can provide MV-CNN a good start point for learning. Furthermore, MV-CNN can still receive supervision from OF-CNN during training by mimicking the output of OF-CNN.

D. Deeply Connected Transfer

We have introduced three strategies in the above sections, namely Teacher Initialization, Supervision Transfer, and their combination. These training methods leverage OF-CNN's knowledge to improve MV-CNN's recognition accuracy. Here, we argue that these three strategies are still kinds of weak knowledge transfer from OF-CNN to MV-CNN. For teacher initialization, knowledge transfer only occurs in the start stage of training. Then MV-CNN will only rely on ground truth label to tune its parameters. For supervision transfer, only the final layer of OF-CNN and MV-CNN is connected. Knowledge that transferred from OF-CNN to MV-CNN is thus very limited, which may impede the performance of MV-CNN. To relieve this shortage, we propose a new strategy, coined as the Deeply Connected Transfer, to perform knowledge transfer in mid-level layers. In this deeply connected transfer approach, we force the feature representation of middle layers of OF-CNN and MV-CNN to be as similar as possible. The insight behind this is that in addition to the final FC layer, middle layers of OF-CNN may also contain useful knowledge that is worth transferring.

In spite of minimizing the difference between $S^n(v)$ and $T^n(o)$ as small as possible, we hope each layer output of student net $S^t(v)$ can approximate the output of the corresponding layer in teacher's net $T^t(o)$, where t represents the ordinal number of each layer. Unlike in the supervision transfer situation that the divergence between two softmax outputs (possibilities) can be measured by cross entropy, we utilize L2 distance to represent the variance between two feature maps. Thus, deeply connected transfer can be seen as a stronger knowledge-transfer strategy than Supervision Transfer. The structure of DTMV-CNN is shown in Figure 4.

L2 loss is utilized to measure the deeply connected supervision loss (DCSL). Thus, DCSL can be defined as:

$$L_{DCSL} = \sum_{t=st}^{n-1} L_2^t(S^t(v), T^t(o)), \tag{5}$$

where st and n stand for ordinal number for one middle layer and the total number of layers respectively. L_2 is the L2 loss function,

$$L_2^t = \frac{1}{k}\sqrt{\sum_{n=1}^{k}(S_k^t(v) - T_k^t(o))^2}, \tag{6}$$

where k is the dimension for layer t of student and teacher net and t is the ordinal number for one specific layer.

As each layer of teacher net provides supervision for student net MV-CNN, deeply connected supervision transfer strategy can fully unleash the supervision ability of teacher net OF-CNN.

For implementation details, we set learning rate to 10^{-3}. The learning rate is reduced to 10^{-4} and 10^{-5} after 50k and 70k iterations respectively. We terminate the learning process at 90k iterations.

V. EXPERIMENTS

In experiment part, we firstly introduce the datasets used in our experiments and then analyze the experimental results.

A. Datasets and Evaluation Protocol

Three datasets are employed to evaluate our proposed real-time action recognition algorithm: UCF101 [9], HMDB51 [10], and THUMOS14 [15]. UCF101 contains 13,320 videos which are divided into three splits for training and testing. We follow the standard setup for three splits and report mean accuracy over three splits.

HMDB51 dataset is among the largest dataset for action recognition. It contains around 7,000 video clips. These clips are split into three sub-datasets. Each contains 3,570 and 1,530 clips for training and testing, respectively. The standard setup for HMDB51 is used and mean accuracies over three splits are reported.

THUMOS14 is a dataset with untrimmed videos. It is originally proposed for action recognition challenge 2014. 13,320 trimmed videos are for training, while another 1,010 and 1,574 untrimmed videos are for validation and testing. Untrimmed videos contain lots of irrelevant frames that make it more challenging in training and testing CNNs. Following [47], both training set and validation set are used for training. We use the official evaluation tool to evaluate performance. According to the standard setup of this dataset, mean Average Precision (mAP) is reported.

For the speed evaluation, the speed is reported as frames per second (fps) on a CPU (E5-2640 v3) and a K40 GPU.

B. Implementation Details

Three data augmentation strategies are used to learn robust CNN features. A 224×224 patch is randomly cropped from training image set. And then random horizontally flipping is implemented to augment training data. Furthermore, following [45],[2] a scale jittering strategy is implemented to help CNN to learn robust features. We set scale ratio for 1, 0.875, and 0.75 to yield a patch of size for 256, 224 and 192, respectively. Then these patches are resized to 224×224. In testing phase, one 224×224 patch cropped from the center of input image are used for evaluation. No data augmentation strategy is used in testing phase.

As HMDB51 is relatively small compared with UCF101 and THUMOS14, we use multi-task learning strategy to train temporal model on HMDB51. Following [1], a CNN is modified to have two separated input layers and two output layers. One is for UCF101 to calculate the loss with ground truth label, while the other is for HMDB51. Furthermore, as the motion vector for the original video in HMDB51 is relatively noisy, we follow [13] to first use ffmpeg to re-encode videos and then extract motion vectors. For other datasets like UCF101 and THUMOS14, we directly extract motion vectors from the original version of videos. Our teacher CNN is trained on TV-L1 optical flow [48] with data augmentation that achieves 81.6% on UCF101 Split1, comparable with the accuracy in the original paper 81.2% [1]. For HMDB51 split1, our teacher CNN achieves 60.0%, which is better than 55.4% on the original two-stream implementation [1]. This significant improvement can be ascribed to the scale jittering strategy.

[2]https://github.com/yjxiong/caffe

TABLE I
ACCURACY OF DIFFERENT CONNECTION STRATEGY FOR DTMV-CNN ON UCF101 SPLIT1

Connection strategy	Accuracy
FC7-FC8	79.3%
FC6-FC8	79.5%
Conv5-FC8	80.0%
Conv4-FC8	80.0%
Conv3-FC8	**80.3%**
Conv2-FC8	80.1%

C. Parameter Sensitivity

We first analyze the parameters for Supervision Transfer method. There are two important parameters existing in Supervision Transfer: temperature *Temp* and weight w for soft target. As suggested in [32], soft target weights are set as $w = Temp^2$ to balance the gradients between two targets. We set temperature *Temp* to 1, 2, and 3 and evaluate them on UCF101 split1. Thus the corresponding soft target weight w is set as 1, 4 and 9. As temperature goes up from 1 to 2, the corresponding accuracy grows up from 79.1% to 79.2%. The accuracy slightly degrade to 79.0% if we set temperature as 3. We can find that accuracies between different temperatures are relatively close, which implies Supervision Transfer strategy is robust for temperature setting. As $Temp = 2$ achieves the best accuracy, we set temperature and weight to 2 and 4 for the following experiment.

Second, we evaluate the accuracy of Deeply Connected Transfer methods and conduct experiments to analyze which layers should be connected for knowledge transfer. Greedy search method is employed to identify the best connection strategy for deeply-transferred motion vector (DTMV). For DTMV, Supervision Transfer and Teacher Initialization are always used. Thus FC8 is connected to TSL loss. Connection for DTMV starts from FC7 layer. The greedy search will be stopped if current connection strategy gets worse accuracy than previous strategy. From Table I, we can observe that the accuracy goes up from 79.3% to 80.3% by connecting FC7 layer to Conv3 layer. As Conv2-FC8 performs slightly worse than Conv3-FC8, we set connection strategy to Conv3-FC8 for further experiments.

For HMDB51 and THUMOS14, we use the same temperature and connection strategy setting as UCF101.

D. Evaluation of MV-CNNs, EMV-CNNs and DTMV-CNNs

This subsection compares and analyzes different knowledge transfer strategies through experiments on UCF101, HMDB51, and THUMOS14. The results are summarized in Table II, Table III, and Table IV. We re-implement two-stream CNNs on this dataset, as Simonyan and Zisserman [1] did not provide results on THUMOS14.

TABLE II

COMPARISON OF TEMPORAL CNN ACCURACY FOR OPTICAL FLOW BASED APPROACH AND MOTION VECTOR BASED METHOD ON UCF101 (SPLIT1). DC REPRESENTS DEEPLY CONNECTED SUPERVISION TRANSFER, ST STANDS FOR SUPERVISION TRANSFER AND TI MEANS TEACHER INITIALIZATION

Temporal CNN	Accuracy
OF-CNN [1] (10 crops)	81.2%
MV-CNN trained from scratch (10 crops)	74.4%
EMV-CNN with ST (10 crops)	77.5%
EMV-CNN with TI (10 crops)	78.2%
EMV-CNN with ST+TI (10 crops)	79.3%
DTMV-CNN with TI+DC (10 crops)	81.0%
DTMV-CNN with ST+TI+DC (10 crops)	80.7%
DTMV-CNN with TI+DC (center crop)	80.3%
DTMV-CNN with ST+TI+DC (center crop)	80.3%

TABLE III

COMPARISON OF TEMPORAL CNN ACCURACY FOR OPTICAL FLOW BASED APPROACH AND MOTION VECTOR BASED METHOD ON HMDB51 (SPLIT1)

Temporal CNN	Accuracy
OF-CNN [1]	55.4%
OF-CNN (Our reimplementation)	60.0%
MV-CNN trained from scratch	45.8%
EMV-CNN with ST+TI	51.2%
DTMV-CNN with ST+TI+DC	**53.0%**

TABLE IV

ACCURACY OF DTMV-CNNS, EMV-CNNS AND MV-CNNS ON THUMOS 14 DATASET. WE ALSO REPORT THE RESULTS OF TWO-STREAM CNNS

CNN	MAP
RGB CNN	57.7%
OF-CNN	55.3%
RGB CNN+OF-CNN	66.1%
MV-CNN	29.8%
EMV-CNN	41.6%
DTMV-CNN	43.6%
RGB CNN+MV-CNN	58.7%
RGB CNN+EMV-CNN	61.5%
RGB CNN+DTMV-CNN	**62.1%**

First, comparing the accuracy of MV-CNN trained from scratch and OF-CNN, we can observe that directly replacing optical flow with motion vector severely degrades the temporal net's accuracy by around 7%, 10% and 25% on UCF101 Split1, HDMB51 Split1 and THUMOS14, respectively. It verifies the fact that block wise motion structure, noisy motion blocks and inaccuracy movement information can severely harm the accuracy of temporal net. Furthermore, we observe that the accuracy gap between MV-CNN and OF-CNN is extremely large in THUMOS14 dataset. As lots of video in THUMOS14 is untrimmed, they contains lots of shots shift and a large number of irrelevant frames which aggravate the difficulties of training MV-CNN.

Second, we can see a significant improvement from MV-CNN to DTMV-CNN for around 6%, 8%, and 14% on UCF101 Split1, HMDB51 Split1, and THUMOS14 respectively, which shows the effectiveness of our proposed deeply connected transfer method. Directly training MV-CNN with ground truth label from scratch lacks elaborate fine-level motion knowledge. Our proposed method shows that although motion vector and optical flow are from different domains, the knowledge of OF-CNN can still be helpful to MV-CNN.

In the next, we study the accuracy of our proposed strategies on UCF101 Split1. For fair comparison with the two stream ConvNets [1], we use data augmentation strategies for testing. Following [1], we use five image crops with the size of 224×224 from 4 corners and the center of a image. We feed forward these crops with their flipped version into CNN. We average these ten image crops to get the final result of the image. From Table II, supervised transfer and teacher initialization outperforms MV-CNN for 3.1% and 3.8%, respectively. Similar to researches in image classification [1], [24], teacher initialization can improve the accuracy of MV-CNN by providing fine-grained knowledge through a pre-trained model. Fine-grained motion knowledge provided by OF-CNN is helpful for training MV-CNN. Furthermore, combining supervision transfer and teacher initialization can further improve the results of teacher initialization about 1.1% on UCF101 Split1, which shows that the softened softmax vector provided by supervision transfer can give richer information than ground truth label solely. As indicated in [32], the softened output of teacher net can be seen as a regulator for MV-CNN to prevent over-fitting. Thus supervision transfer can help to train better MV-CNN. We observe that deeply-transferred motion vector (DTMV-CNN) can further boost the results of EMV-CNN for 1%, 1.8% and 2% on UCF101, HMDB51, and THUMOS14 respectively. It indicates that knowledge contained in each layer of OF-CNN is useful for MV-CNN to enhance its generalization ability and only using the final fully connection layer (FC8) as supervision can not fully utilize the knowledge of OF-CNN. We noticed that DTMV with TI+DC and ST+TI+DC achieves similar performance. They all surpass the performance of EMV-CNN, which indicate that, for knowledge transfer, deeply connected transfer can provide much stronger supervision to MV-CNN than supervision transfer.

We also show the accuracy of DTMV-CNN with 10 crops and the one with center crop on UCF101 Split1 in Table II. DTMV-CNN with center crop shows similar performance with the one with 10 crops. Thus in the HMDB51 and THUMOS14 dataset, only one 224×224 patch cropped from the center of input image for testing.

Furthermore, we can observe that combining temporal net DTMV-CNN and spatial net can outperform MV-CNN with RGB net. It indicates that the knowledge of DTMV-CNN is more complementary to spatial net than MV-CNN.

Finally, we analyze each category accuracy of MV-CNNs, EMV-CNNs and DTMV-CNNs on UCF101 Split1. According to the class category in [9], videos in UCF101 can be classified into 101 different classes. These classes can be categorized as 4 major categories: Human-Human action, Human-Object action, Human-Instrument action, and Human-Sports action. The comparison for accuracy on each category can be seen from Fig. 6. EMV-CNN shows significant improvement over MV-CNN by around 4%, 6%, and 4% on categories of Human-Human action, Human-Object action and Human-Sports

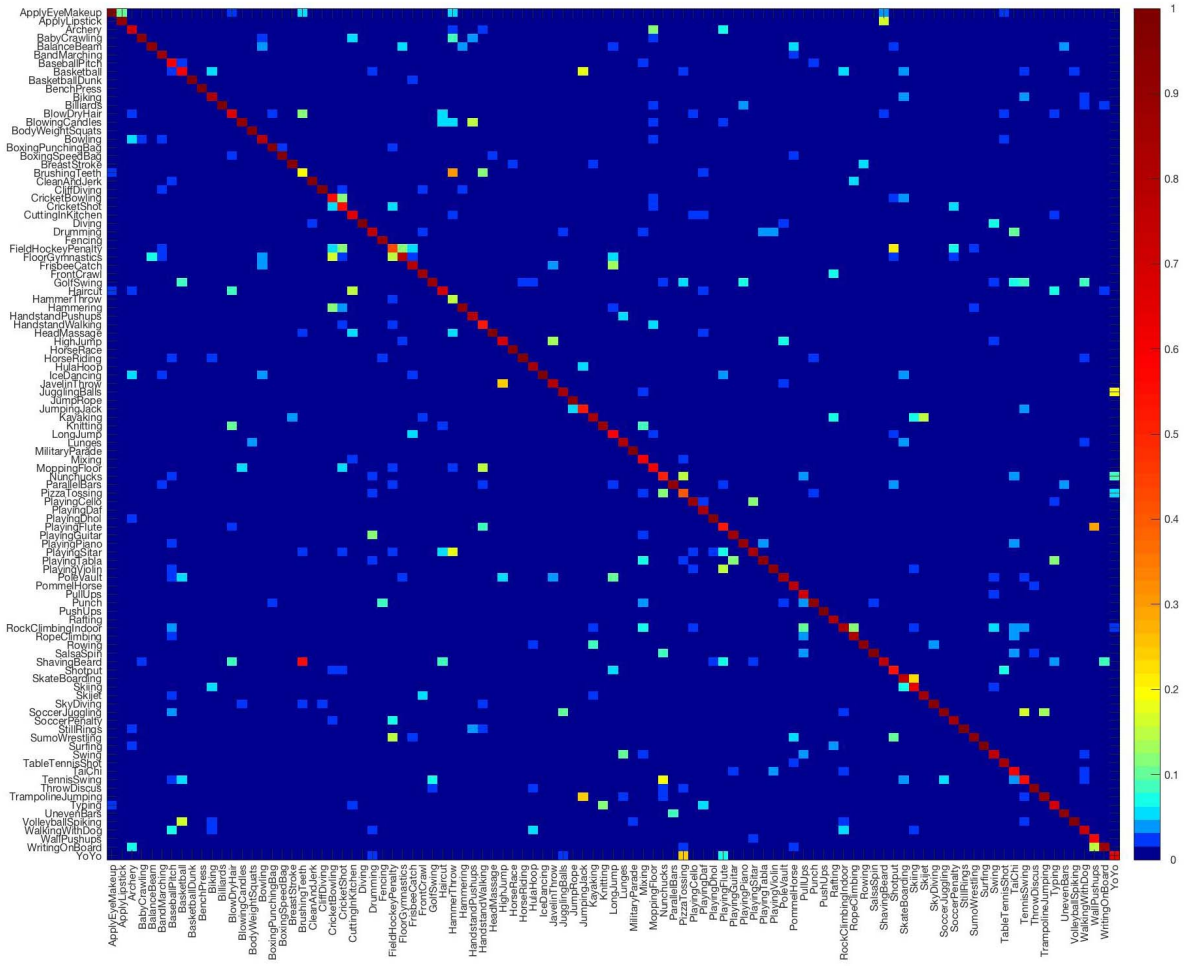


Fig. 5. Confusion matrix for DTMV-CNN of 101 classes on UCF101 (split1).

action, respectively. It indicates that the strategies of transferring knowledge from optical flow CNN to motion vector CNN can improve the motion vector CNN's performance. However for the Human-Instrument category, EMV-CNN shows similar accuracy with MV-CNN. As videos in Human-Instrument category require detail motion information, similar to MV-CNN, EMV-CNN may still lack knowledge for fine-grained motion. By using the proposed deeply connected transfer method, DTMV-CNN further improves the results of EMV-CNN on categories for Human-Human action and Human-Instrument action by 2% and 4%, respectively. It shows that the purposed method transfers more knowledge of detail motion to DTMV-CNN than EMV-CNN.

E. Confusion Matrix for DTMV-CNN

We show the confusion matrix for DTMV-CNN in Fig. 5. It can be shown that DTMV-CNN performs well in most videos for Human-Human action category like *BandMarching* and *HeadMassage*. However, DTMV-CNN performs worse in class *BrushingTeeth*. For *BrushingTeeth* , DTMV-CNN misclassifies majority of videos into *ShavingBeards*. It may be due to the action in *BrushingTeeth* is similar to the one in *ShavingBeards*.

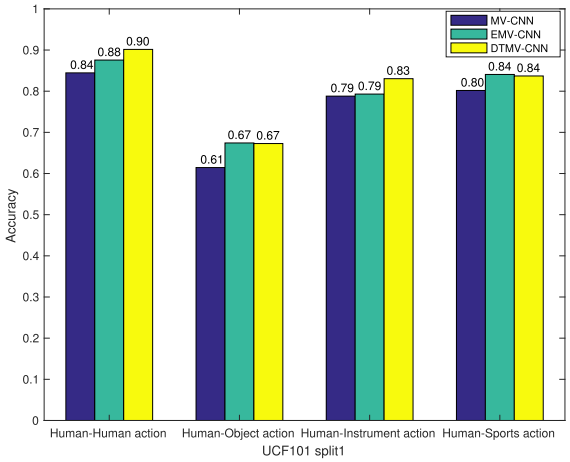


Fig. 6. Performance of MV-CNN, EMV-CNN and DTMV-CNN on four categories of UCF101 Split1.

F. Speed Evaluation

We analyze the speed of different components of our proposed approach. In our implementation, motion vector and RGB image are extracted with CPU, while GPU is utilized

TABLE V

SPEED OF EACH COMPONENTS IN REAL-TIME ACTION RECOGNITION SYSTEM. MV AND RGB STANDS FOR MOTION VECTOR AND RGB IMAGE EXTRACTION, WHILE CNN MEANS CONVOLUTIONAL NEURAL NETWORK PROCESSING

Dataset	MV and RGB (fps)	CNN (fps)	Total (fps)
UCF101	675.7	925.9	390.7
HMDB51	675.7	925.9	390.7
THUMOS14	757.6	925.9	403.2

TABLE VI

COMPARISON OF SPEED FOR OPTICAL FLOW FIELDS AND MOTION VECTORS. MV MEANS MOTION VECTOR

Dataset	Spatial Resolution	Brox's Flow[11] (GPU) (fps)	MV (CPU) (fps)
UCF101	320×240	16.7	675.7
HMDB51	320×240	16.7	675.7
THUMOS14	320×180	17.5	757.6

to process the feed forward calculation of RGB CNN and MV-CNN. As the I/O is related to hardware and operating system, the computation time reported doesn't include I/O. However time for video reading and decompression is still included in processing time. A volume with 10 frames of motion vector images and one RGB image is processed at each time. The speed is measured based on the time cost on each frame instead of each volume.

First, speeds of each component on UCF101, HMDB51, and THUMOS14 are shown in Table V. We evaluate the speeds for different spatial resolutions. For UCF101 and HMDB51, the spatial resolution is 320×240, while THUMOS14's video has a resolution of 320×180. The calculation of our approach includes video reading and decoding, extraction for motion vector and RGB, and CNN processing. Around 57% time is used for motion vector and RGB image extraction with CPU. The rest of time is occupied by the feed forward computation of RGB-CNN and MV-CNN on GPU. As GPU is optimized for matrix multiplication, feed forward calculating of CNN is slightly faster than extracting motion vector and RGB images from video. We can observe that the total processing speed for UCF101, HMDB51, and THUMOS14 is 390.7, 390.7 and 403.2 fps, respectively, which is one order faster than real-time processing requirement (25 fps).

Second, speeds for motion vectors and optical flow (Brox's flow) extraction are compared in Table VI. Although the processing speed of motion vector is slight different under different resolutions, motion vector extraction is almost 40 times faster than optical flow calculation. Even compared with real-time requirement (25 fps), extraction of motion vector is almost 27 times faster. For two-stream CNNs, around 85% of processing time is occupied by optical flow calculation, which prohibits the classical two-stream approach to be conducted in real-time.

G. Comparison With the State of the Art

We compare our proposed method with several state-of-the-art methods in this subsection. The results are summarized in Tables VII–IX. SVM is employed in most state-of-the-art approaches [1], [2], [13] for classification. Different to these step-wise methods, our proposed real-time action recognition method is an end-to-end approach.

TABLE VII

COMPARISON OF SPEED AND ACCURACY WITH STATE-OF-THE-ART ON UCF101

	Accuracy	FPS
MV+FV (CPU) (re-implement) [13]	78.5%	132.8
C3D (1 net) (GPU) [4]	82.3%	313.9
C3D (3 net) (GPU) [4]	85.2%	-
iDT+FV (CPU) [2]	85.9%	2.1
Two-stream CNNs (GPU) [1]	88.0%	14.3
EMV+RGB-CNN (GPU) [14]	86.4%	**390.7**
TSN (GPU) [8]	94.0%	< 25
DTMV+RGB-CNN	**87.5%**	**390.7**

TABLE VIII

COMPARISON OF SPEED AND ACCURACY WITH STATE-OF-THE-ART ON HMDB51

	Accuracy	FPS
MV+FV (CPU) [13]	46.7%	101.0
MV+VLAD (CPU) [13]	46.3%	227.8
iDT+FV (CPU) [2]	57.2%	2.1
Two-stream CNNs (GPU) [1]	59.4%	14.3
TSN (GPU) [8]	68.5%	< 25
DTMV+RGB-CNN	**55.3%**	**390.7**

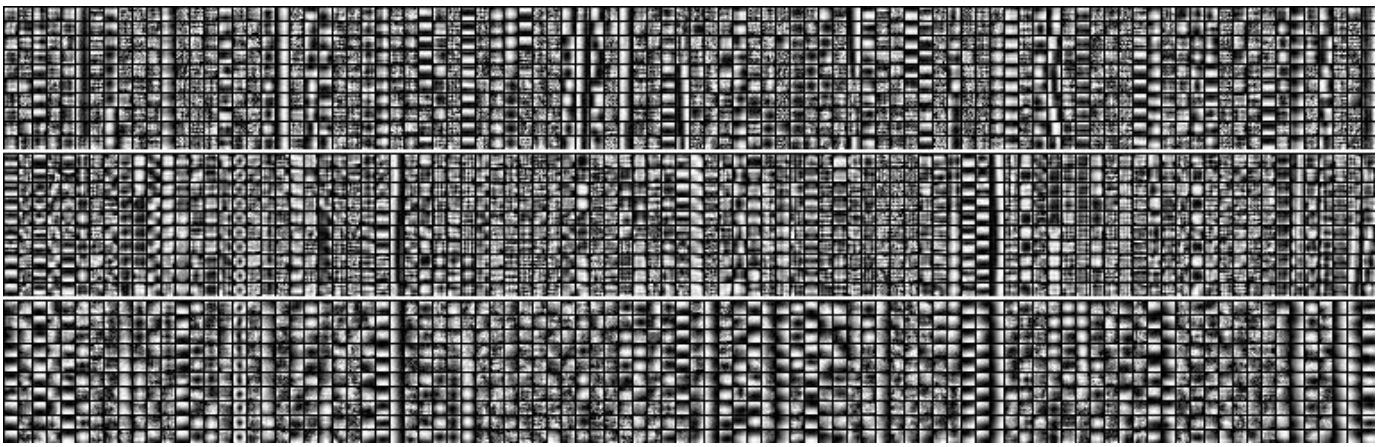
TABLE IX

COMPARISON OF SPEED AND ACCURACY WITH STATE-OF-THE-ART ON THUMOS14

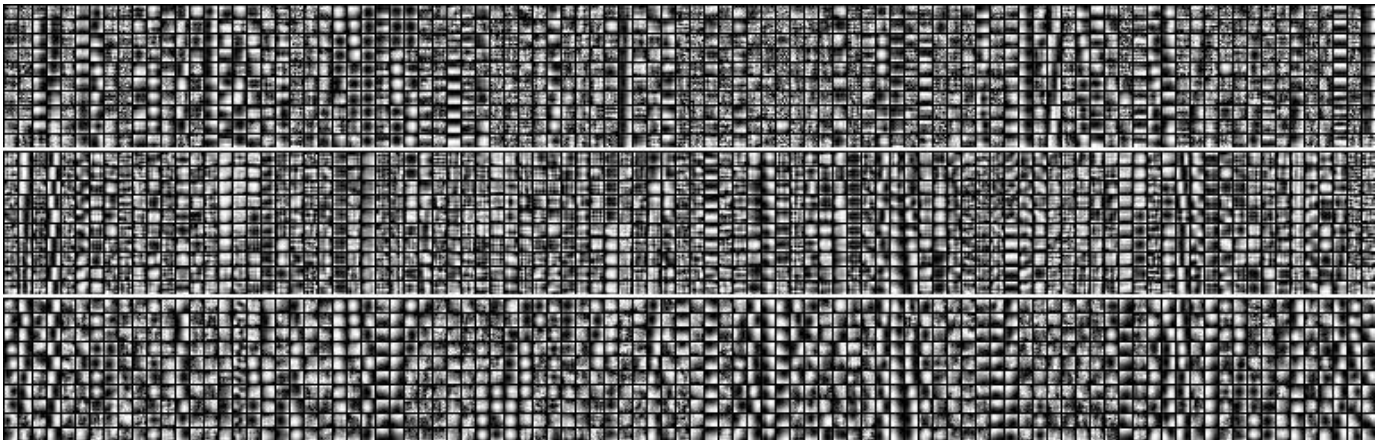
	Accuracy	FPS
Objects (GPU) [47]	44.7%	-
iDT+CNN (CPU+GPU) [49]	62.0%	< 2.38
Motion (iDT+FV) (CPU) [47]	63.1%	2.38
Objects+Motion (CPU+GPU) [47]	71.6%	< 2.38
UntrimmedNet (GPU) [50]	82.2%	< 25
TSN (GPU) [51]	80.1%	< 25
EMV+RGB-CNN (GPU) [14]	61.5%	403.2
DTMV+RGB-CNN	**62.1%**	**403.2**

First, speed and accuracy performance on UCF101 (3 Splits) are analyzed. As Kantorov and Laptev [13] did not report accuracy on this dataset, we re-implement their methods using the public code offered by Kantorov and Laptev [13]. From Table VII, we can observe that DTMV+RGB-CNN is around 3 times faster than previous action recognition research on motion vector [13]. Although it may not be a fair comparison as MV+FV only uses CPU, DTMV+RGB-CNN still outperforms MV+FV for 9%. Compared with optical flow based approaches [1], [2], [8], our approach is 180 times, 27 times, and 16 times faster than iDT+FV, classical Two-stream CNNs, and TSN respectively. Our approach also achieves higher performance than iDT+FV and achieves similar performance with Two-stream CNNs. Although TSN achieves impressive results on UCF101 dataset, it is based on optical flow. Thus, TSN cannot be conducted in real time. We also compare our approach with RGB-based algorithm [4]. Our approach achieves better accuracy and faster processing speed than C3D (1 net) and C3D (3 net).

Second, we compare the performance on HMDB51 (3 Splits). Our method significantly outperforms other motion vector based algorithm [13] by around 9%. At the same time, our approach is 100 fps faster than MV+VLAD.



(a)



(b)

Fig. 7. Filters for Conv1 layer of MV-CNN, DTMV-CNN, and OF-CNN. (a) Filters on horizontal components of Conv1 layer for MV-CNN, DTMV-CNN, and OF-CNN. From top to down: MV-CNN, DTMV-CNN, and OF-CNN. (b) Filters on vertical components of Conv1 layer for MV-CNN, DTMV-CNN, and OF-CNN. From top to down: MV-CNN, DTMV-CNN, and OF-CNN.

Compared with optical flow performance, our approach shows slightly worse accuracy than iDT [2] and classical two-stream CNNs [1]. TSN achieves superior performance than us. However, as these methods requires optical flow as input, our method is around 300 fps much faster than iDT, two-stream CNNs, and TSN.

Finally, results on THUMOS14 dataset (Table IX) are compared. Our approach achieves better performance than RGB based approach (Objects) [47] by 18%. For optical flow based approach, DTMV+RGB-CNN is slightly better than iDT+CNN and shows comparable performance with iDT+FV, but exhibits worse result than Objects+Motion. However, as Objects+Motion and iDT+FV need to calculate optical flow, even with efficient implementation, optical flow based approach is one order slower than real-time requirement. Our method is around 16 times faster than real-time processing (25 fps) and exhibits 200 times quicker than optical flow based algorithm [11].

H. Visualization of Filters

To verify the effectiveness of our proposed training strategy, filters of both horizontal and vertical components of the first layer (Conv1) for MV-CNN, DTMV-CNN, and OF-CNN are visualized in Fig. 7. As analyzed before, motion vector lacks fine-grained motion information and contains imprecise motion blocks. We can clearly observe that filters for MV-CNN are much coarser than those of OF-CNN and exhibit more noisy part. On the other hand, filters of OF-CNN show clear and smooth boundaries which are more suitable for extracting effective features from videos. We hope that the knowledge of OF-CNN can be transferred to MV-CNN. However, due to the coarse nature of motion vector, the fine features learned by OF-CNN can not be directly applied into MV-CNN. Thus, at the same time, we also hope that MV-CNN can benefit the knowledge from both optical flow domain and motion vector domain. From the filters of DTMV-CNN, we can observe that the filters are smoother than those in MV-CNN but still contain certain coarse structures which are learned from motion vector. This indicates that our proposed method can transfer OF-CNN knowledge to DTMV-CNN. Our experimental results on various datasets also verify that our training strategies help to enhance motion vector CNN with better generalization abilities.

VI. Conclusion

In this paper, motion vector CNN has been proposed to accelerate the processing speed of deep learning approach for action recognition. As motion vectors are already encoded

in video stream, it can be directly extracted without extra computation. However, as motion vectors only contain block level and inaccurate motion information, directly training CNN from scratch with motion vector severally degrades the action recognition performance. To tackle this problem, DTMV-CNN is proposed to enable knowledge transfer from optical flow domain to motion vector domain. Performance of DTMV-CNN on three challenging datasets verify the effectiveness of our training approach which shows significantly better performance than MV-CNN trained from scratch. Furthermore, our proposed real-time action recognition approach is around 16 times faster than real-time requirement and achieves 391 fps, 391 fps, and 403 fps on UCF101, HMDB51 and THUMOS2014 with high performance.

REFERENCES

[1] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Proc. NIPS*, Montreal, Canada, Dec. 2014, pp. 568–576.

[2] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *Proc. ICCV*, Sydney, NSW, Australia, Mar. 2013, pp. 3551–3558.

[3] L. Wang, Y. Qiao, and X. Tang, "MoFAP: A multi-level representation for action recognition," *Int. J. Comput. Vis.*, vol. 119, no. 3, pp. 254–271, Sep. 2016.

[4] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4489–4497.

[5] L. Wang, Y. Qiao, and X. Tang, "Action recognition with trajectory-pooled deep-convolutional descriptors," in *Proc. CVPR*, Boston, MA, USA, Jun. 2015, pp. 4305–4314.

[6] J. Sivic and A. Zisserman, "Video Google: A text retrieval approach to object matching in videos," in *Proc. ICCV*, Nice, France, Apr. 2003, pp. 1470–1477.

[7] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek, "Image classification with the fisher vector: Theory and practice," *Int. J. Comput. Vis.*, vol. 105, no. 3, pp. 222–245, Dec. 2013.

[8] L. Wang *et al.*, "Temporal segment networks: Towards good practices for deep action recognition," in *Proc. ECCV*, Amsterdam, The Netherlands, Oct. 2016, pp. 20–36.

[9] K. Soomro, A. R. Zamir, and M. Shah, "UCF101: A dataset of 101 human actions classes from videos in the wild," *CoRR*, pp. 1–7, Nov. 2012.

[10] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "HMDB: A large video database for human motion recognition," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Nov. 2011, pp. 2556–2563.

[11] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, May 2004, pp. 25–36.

[12] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Techn.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.

[13] V. Kantorov and I. Laptev, "Efficient feature extraction, encoding, and classification for action recognition," in *Proc. CVPR*, Columbus, OH, USA, Sep. 2014, pp. 2593–2600.

[14] B. Zhang, L. Wang, Z. Wang, Y. Qiao, and H. Wang, "Real-time action recognition with enhanced motion vector cnns," in *Proc. CVPR*, Las Vegas, NV, USA, Jun. 2016, pp. 2718–2726.

[15] Y.-G. Jiang *et al.* (2014). *THUMOS Challenge: Action Recognition With a Large Number of Classes*. [online]. Available: http://crcv.ucf.edu/THUMOS14/

[16] T. Leung and J. Malik, "Representing and recognizing the visual appearance of materials using three-dimensional textons," *Int. J. Comput. Vis.*, vol. 43, no. 1, pp. 29–44, Jun. 2001.

[17] I. Laptev, "On space-time interest points," *Int. J. Comput. Vis.*, vol. 64, nos. 2–3, pp. 107–123, 2005.

[18] H. Wang, M. M. Ullah, A. Kläser, I. Laptev, and C. Schmid, "Evaluation of local spatio-temporal features for action recognition," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, London, U.K., Sep. 2009, pp. 124.1–124.11.

[19] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, "Action recognition by dense trajectories," in *Proc. CVPR*, Colorado Springs, CO, USA, Aug. 2011, pp. 3169–3176.

[20] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. CVPR*, San Diego, CA, USA, Jul. 2005, pp. 886–893.

[21] N. Dalal, B. Triggs, and C. Schmid, "Human detection using oriented histograms of flow and appearance," in *Proc. ECCV*, Graz, Austria, May 2006, pp. 428–441.

[22] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *Proc. CVPR*, San Francisco, CA, USA, Aug. 2010, pp. 3304–3311.

[23] X. Peng, L. Wang, X. Wang, and Y. Qiao, "Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice," *Comput. Vis. Image Understand.*, vol. 150, pp. 109–125, Sep. 2016.

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. 25th Int. Conf. Neural Inf. Process. Syst.*, Lake Tahoe, Nevada, Dec. 2012, pp. 1097–1105.

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, Las Vegas, NV, USA, Jun. 2016, pp. 770–778.

[26] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proc. CVPR*, Columbus, OH, USA, Sep. 2014, pp. 1725–1732.

[27] L. Wang, W. Li, W. Li, and L. Van Gool, "Appearance-and-relation networks for video classification," *CoRR*, pp. 1–12, Nov. 2017.

[28] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *Proc. CVPR*, Las Vegas, NV, USA, Jun. 2016, pp. 1933–1941.

[29] C. Feichtenhofer, A. Pinz, and R. Wildes, "Spatiotemporal residual networks for video action recognition," in *Proc. NIPS*, Barcelona, Spain, Dec. 2016, pp. 3468–3476.

[30] J. Y.-H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classification," in *Proc. CVPR*, Boston, MA, USA, Oct. 2015, pp. 4694–4702.

[31] Z. Wu, Y.-G. Jiang, J. Wang, J. Pu, and X. Xue, "Exploring inter-feature and inter-class relationships with deep neural networks for video classification," in *Proc. ACM Int. Conf. Multimedia*, Orlando, FL, USA, Nov. 2014, pp. 167–176.

[32] G. E. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *CoRR*, pp. 1–9, Mar. 2015.

[33] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks: Training deep neural networks with weights and activations constrained to $+1$ or -1," *CoRR*, pp. 1–11, Mar. 2016.

[34] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi, "XNOR-Net: ImageNet classification using binary convolutional neural networks," in *Proc. ECCV*, Amsterdam, The Netherlands, Oct. 2016, pp. 525–542.

[35] V. Vapnik and R. Izmailov, "Learning using privileged information: Similarity control and knowledge transfer," *J. Mach. Learn. Res.*, vol. 16, pp. 2023–2049, Sep. 2015.

[36] M. Lapin, M. Hein, and B. Schiele, "Learning using privileged information: SVM+ and weighted SVM," *Neural Netw.*, vol. 53, pp. 95–108, May 2014.

[37] D. Lopez-Paz, L. Bottou, B. Schölkopf, and V. Vapnik, "Unifying distillation and privileged information," *CoRR*, pp. 1–9, Feb. 2016.

[38] W. Li, L. Niu, and D. Xu, "Exploiting privileged information from Web data for image categorization," in *Proc. ECCV*, Zurich, Switzerland, Sep. 2014, pp. 437–452.

[39] L. Niu, W. Li, and D. Xu, "Exploiting privileged information from Web data for action and event recognition," *Int. J. Comput. Vis.*, vol. 118, no. 2, pp. 130–150, Jun. 2016.

[40] S. Gupta, J. Hoffman, and J. Malik, "Cross modal distillation for supervision transfer," in *Proc. CVPR*, Las Vegas, NV, USA, Jun. 2016, pp. 2827–2836.

[41] L. Wang, S. Guo, W. Huang, Y. Xiong, and Y. Qiao, "Knowledge guided disambiguation for large-scale scene classification with multi-resolution CNNs," *IEEE Trans. Image Process.*, vol. 26, no. 4, pp. 2055–2068, Apr. 2017.

[42] L. Wang, Z. Wang, Y. Qiao, and L. Van Gool, "Transferring deep object and scene representations for event recognition in still images," *Int. J. Comput. Vis.*, pp. 1–20, Sep. 2017.

[43] G. Farnebäck, "Two-frame motion estimation based on polynomial expansion," in *Proc. SCIA*, Halmstad, Sweden, Jun. 2003, pp. 363–370.
[44] B. Bross, W.-J. Han, J.-R. Ohm, G. J. Sullivan, Y.-K. Wang, and T. Wiegand, "High efficiency video coding (HEVC) text specification draft 10," in *Proc. JCT-VC*, 2013, p. 91.
[45] L. Wang, Y. Xiong, Z. Wang, and Y. Qiao, "Towards good practices for very deep two-stream ConvNets," *CoRR*, pp. 1–5, Jul. 2015.
[46] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, pp. 1–14, Apr. 2015.
[47] M. Jain, J. C. van Gemert, and C. G. M. Snoek, "What do 15,000 object categories tell us about classifying and localizing actions?" in *Proc. CVPR*, Boston, MA, USA, Jun. 2015, pp. 46–55.
[48] J. S. Pérez, E. Meinhardt-Llopis, and G. Facciolo, "TV-L1 optical flow estimation," *IPOL Image Process. Line*, vol. 3, pp. 137–150, Jul. 2013.
[49] L. Wang, Y. Qiao, and X. Tang, "Action recognition and detection by combining motion and appearance features," in *Proc. THUMOS Action Recognit. Challenge*, 2014, p. 2.
[50] L. Wang, Y. Xiong, D. Lin, and L. Van Gool, "Untrimmednets for weakly supervised action recognition and detection," in *Proc. CVPR*, Honolulu, HI, USA, Jul. 2017, pp. 4325–4334.
[51] L. Wang *et al.*, "Temporal segment networks for action recognition in videos," *CoRR*, pp. 1–14, May 2017.

Bowen Zhang received the B.Eng. degree in computer science and technology and the M.Eng. degree in computer technology from Tongji University, Shanghai, China, in 2014 and 2017, respectively. He is currently pursuing the Ph.D. degree with the Department of Computer Science, University of Southern California, Los Angeles, USA. His research interests include computer vision and machine learning. He was the winner at the ActivityNet Large Scale Activity Recognition Challenge 2016 in video classification and the first runner-up at the ActivityNet Large Scale Activity Recognition Challenge 2017 in untrimmed video classification, trimmed video classification, and temporal action localization.

Limin Wang received the B.Sc. degree from Nanjing University, Nanjing, China, in 2011, and the Ph.D. degree from The Chinese University of Hong Kong, Hong Kong, in 2015. He is currently a Post-Doctoral Researcher with the Computer Vision Laboratory, ETH Zurich. His research interests include computer vision and deep learning. He was the first runner-up at the ImageNet Large Scale Visual Recognition Challenge 2015 in scene recognition, the winner at the ActivityNet Large Scale Activity Recognition Challenge 2016 in video classification, and the first runner-up at the ActivityNet Large Scale Activity Recognition Challenge 2017 in untrimmed video classification, trimmed video classification, and temporal action localization.

Zhe Wang received the B.Eng. degree from the Beijing University of Posts and Telecommunications, Beijing, China, in 2014. He is currently pursuing the Ph.D. degree with the University of California at Irvine, Irvine, USA. He was a Research Assistant with The Chinese University of Hong Kong and the Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, from 2014 to 2016. His current research interests include computer vision and deep learning. He was the winner of the ChaLearn Challenge 2015 in Action/Interaction Recognition and Culture Event Classification and at the ActivityNet Large Scale Activity Recognition Challenge 2016 in video classification.

Yu Qiao (SM'13) received the Ph.D. degree from the University of Electro-Communications, Japan, in 2006. He was a JSPS Fellow and a Project Assistant Professor with The University of Tokyo from 2007 to 2010. He is currently a Professor with the Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences. He has authored over 140 papers in journals and conference including, TPAMI, IJCV, TIP, ICCV, CVPR, ECCV, and AAAI. His research interests include computer vision, deep learning, and intelligent robots. He received the Lu Jiaxi Young Researcher Award from the Chinese Academy of Sciences in 2012. He was the first runner-up at the ImageNet Large Scale Visual Recognition Challenge 2015 in scene recognition and the winner at the ActivityNet Large Scale Activity Recognition Challenge 2016 in video classification.

Hanli Wang (M'08–SM'12) received the B.E. and M.E. degrees in electrical engineering from Zhejiang University, Hangzhou, China, in 2001 and 2004, respectively, and the Ph.D. degree in computer science from the City University of Hong Kong, Kowloon, Hong Kong, in 2007. From 2007 to 2008, he was a Research Fellow with the Department of Computer Science, City University of Hong Kong. From 2007 to 2008, he was a Visiting Scholar with Stanford University, Palo Alto, CA, USA. From 2008 to 2009, he was a Research Engineer with Precoad Inc., Menlo Park, CA, USA. From 2009 to 2010, he was an Alexander von Humboldt Research Fellow with the University of Hagen, Hagen, Germany. Since 2010, he has been a Full Professor with the Department of Computer Science and Technology, Tongji University, Shanghai, China. His current research interests include digital video coding, computer vision, and machine learning.