Temporal Segment Networks for Action Recognition in Videos

Limin Wang, *Member, IEEE*, Yuanjun Xiong, Zhe Wang, Yu Qiao, *Senior Member, IEEE*, Dahua Lin, Xiaoou Tang, *Fellow, IEEE*, and Luc Van Gool, *Member, IEEE*

Abstract—We present a general and flexible video-level framework for learning action models in videos. This method, called temporal segment network (TSN), aims to model long-range temporal structure with a new segment-based sampling and aggregation scheme. This unique design enables the TSN framework to efficiently learn action models by using the whole video. The learned models could be easily deployed for action recognition in both trimmed and untrimmed videos with simple average pooling and multi-scale temporal window integration, respectively. We also study a series of good practices for the implementation of the TSN framework given limited training samples. Our approach obtains the state-the-of-art performance on five challenging action recognition benchmarks: HMDB51 (71.0 percent), UCF101 (94.9 percent), THUMOS14 (80.1 percent), ActivityNet v1.2 (89.6 percent), and Kinetics400 (75.7 percent). In addition, using the proposed RGB difference as a simple motion representation, our method can still achieve competitive accuracy on UCF101 (91.0 percent) while running at 340 FPS. Furthermore, based on the proposed TSN framework, we won the video classification track at the ActivityNet challenge 2016 among 24 teams.

Index Terms—Action recognition, temporal segment networks, temporal modeling, good practices, ConvNets

1 Introduction

Video-based action recognition has drawn considerable attention from the academic community [1], [2], [3], [4], [5], owing to its applications in many areas like security and behavior analysis. For action recognition in videos, there are two crucial and complementary cues: appearance and temporal dynamics. The performance of a recognition system depends, to a large extent, on whether it is able to extract and utilize relevant information therefrom. However, extracting such information is non-trivial due to a number of difficulties, such as scale variations, view point changes, and camera motions. Thus it becomes crucial to design effective representations to tackle these challenges while learning categorical information of action classes.

- *L. Wang is with State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China. E-mail: lmwang.nju@gmail.com.*
- *Y. Xiong is with Amazon Web Services, Seattle, WA 98101. E-mail: yuanjx@amazon.com.*
- *Z. Wang is with the Department of Computer Science, University of California at Irvine, Irvine, CA 92697. E-mail: buptwangzhe2012@gmail.com.*
- *Y. Qiao is with Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China. E-mail: yu.qiao@siat.ac.cn.*
- *D. Lin and X. Tang are with the Department of Information Engineering, The Chinese University of Hong Kong, Shatin, Hong Kong. E-mail: {dhlin, xtang}@ie.cuhk.edu.hk.*
- *L. Van Gool is with the Computer Vision Laboratory, ETH Zurich, Zurich 8092, Switzerland. E-mail: vangool@vision.ee.ethz.ch.*

Manuscript received 8 May 2017; revised 10 Aug. 2018; accepted 19 Aug. 2018. Date of publication 2 Sept. 2018; date of current version 10 Oct. 2019. (Corresponding author: Limin Wang.)
Recommended for acceptance by I. Laptev.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TPAMI.2018.2868668

Recently, Convolutional Neural Networks (ConvNets) [6] have achieved great success in classifying images of objects [7], [8], [9], scenes [10], [11], [12], and complex events [13], [14]. ConvNets have also been introduced to solve the problem of video-based action recognition [1], [15], [16], [17]. Deep ConvNets come with excellent modeling capacity and are capable of learning discriminative representations from raw visual data in large-scale supervised datasets (e.g., ImageNet [18], Places [10]). However, unlike image classification, improvement brought by end-to-end deep ConvNets remains limited compared with traditional hand-crafted features for video-based action recognition.

We argue that the application of ConvNets to action recognition in unconstrained videos is impeded by three major obstacles. First, although long-range temporal structure has been proven crucial for understanding the dynamics in traditional methods [19], [20], [21], [22], [23], it has not been considered as a critical factor in deep ConvNet frameworks [1], [15], [16], [17]. These methods usually focus on appearances and short-term motions (i.e., up to 16 frames), thus lacking the capacity to incorporate long-range temporal structure. Recently there are a few attempts [4], [24], [25] to deal with this problem. These methods mostly rely on dense temporal sampling with a pre-defined sampling interval, which would incur excessive computational cost when applied to long videos. More importantly, the limited memory space available severely limits the duration of video to be modeled. This poses a risk of missing important information for videos longer than the affordable sampling duration.

Second, existing action recognition methods were mostly devised for trimmed videos. However, to deploy the learned action models in a realistic setting, we often need to deal with untrimmed videos (e.g., THUMOS [26], ActivityNet [27]),

where each action instance may only occupy a small portion of the whole video. The dominating background portions may interfere with the prediction of action recognition models. To mitigate this issue, we need to take account of focusing on action instances and avoiding the influence of background video at the same time. Therefore, it is a non-trivial task to apply the learned action models to action recognition in untrimmed videos.

Third, training action recognition models often meets a number of practical difficulties: 1) training deep ConvNets usually requires a large volume of training samples to achieve optimal performance. However, publicly available action recognition datasets (e.g., UCF101 [28], HMDB51 [29]) remain limited in both size and diversity, making the model training prone to over-fitting. 2) optical flow extraction to capture short-term motion information becomes a computational bottleneck for deploying the learned models to large-scale action recognition datasets.

These challenges motivate us to study the action recognition problem in this paper from the following three aspects: *1) how to effectively learn video representation that captures long-range temporal structure; 2) how to exploit these learned ConvNet models for the more realistic setting of untrimmed videos; 3) how to efficiently learn the ConvNet models given limited training samples and apply them on large scale data.*

To capture long-range temporal structure, we develop a modular video-level architecture, called *temporal segment network* (TSN), which provides a conceptually simple, flexible, and general framework for learning action models in videos. It is based on our observation that *consecutive frames are highly redundant, where a sparse and global temporal sampling strategy would be more favorable and efficient in this case.* The TSN framework first extracts short snippets over a long video sequence with a sparse sampling scheme, where the video is first divided into a fixed number of segments and one snippet is randomly sampled from each segment. Then, a segmental consensus function is employed to aggregate information from the sampled snippets. By this means, temporal segment networks can model long-range temporal structures over the whole video, in a way that its computational cost is independent of the video duration. In practice, we comprehensively study the effect of different segment numbers and propose five aggregation functions to summarize the prediction scores from these sampled snippets, including three basic forms: average pooling, max pooling, and weighted average, as well as two advanced schemes: top-$\mathcal{K}$ pooling and adaptive attention weighting. The latter two are designed to automatically highlight discriminative snippets while reducing the impact of less relevant ones during training, thus contribute to a better learned action model.

To apply the action models learned by TSN to untrimmed videos, we design a hierarchical aggregating strategy, called Multi-scale Temporal Window Integration (M-TWI), to yield the final prediction results for untrimmed videos. Most of previous action recognition methods are constrained to classify manually trimmed video clips. However, this setting may be impractical and unrealistic, as videos on the web are untrimmed by nature and manually trimming these videos is labor demanding. Following the idea of temporal segment network framework, we first divide the untrimmed video into a sequence of short windows of fixed duration. We then perform action recognition for each window independently by max pooling over these snippet-level recognition scores inside this window. Finally, following the aggregation function of temporal segment network framework, we employ the top-$\mathcal{K}$ pooling or attention weighting to aggregate the predictions from these windows to produce the video-level recognition results. Due to its capability of implicitly selecting intervals with discriminative action instances while suppressing the influence of noisy background, this newly designed aggregation module is effective for untrimmed video recognition

To tackle the practical difficulties in learning and applying action recognition models, we discover a number of good practices to resolve the issues caused by the limited training samples, and perform a systematical study over the input modalities to unleash the full potential of ConvNets for action recognition. Specifically, we first propose a *cross-modality initialization* strategy to transfer the learned representations from RGB modality to other modalities like optical flow and RGB difference. Second, we develop a principled method to perform Batch Normalization (BN) in a fine-tuning scenario, denoted as *partial BN*, where only the mean and variance of first BN layer are updated adaptively to handle domain shift. Moreover, to fully utilize visual content from videos, we empirically study four types of input modalities with our temporal segment network framework, namely a single RGB image, stacked RGB difference, stacked optical flow field, and stacked warped optical flow field. Combining RGB and RGB difference, we build the best-ever real-time action recognition system, which has numerous potential applications in real-world problems.

We perform experiments on five challenging action recognition datasets, namely HMDB51 [29], UCF101 [28], THUMOS [26], ActivityNet [27], and Kinetics [30], to verify the effectiveness of our method for action recognition in both trimmed and untrimmed videos. In experiments, models learned using the TSN framework outperform the state of the art on these five challenging action recognition benchmark datasets. Additionally, following the general TSN framework, we further improve our action recognition method by introducing the latest deep model architectures (e.g., ResNet [31] and Inception V3 [32]), and incorporating the audio as a complementary channel. Our final action recognition method secures the 1st place in untrimmed video classification at the ActivityNet Large Scale Activity Recognition Challenge 2016. We also visualize our learned two-stream models trying to provide insights into how they work. These visualized models also justify the effectiveness of the TSN framework qualitatively.

Overall, we analyze different aspects of the problems in efficiently and effectively learning and applying action recognition models and make *three major contributions*: 1) we propose an end-to-end framework, dubbed temporal segment network, for learning video representation that captures long-term temporal information; 2) we design a hierarchical aggregation scheme to apply action recognition models to untrimmed videos; 3) we investigate a series of good practices for learning and applying deep action recognition models.

This journal paper extends our previous work [33] in a number of aspects. *First*, we introduce new aggregation

functions into the TSN framework, which turn out to be effective to highlight important snippets while suppress background noise. *Second*, we extend the original action recognition pipeline to untrimmed video classification, by designing a hierarchical aggregating strategy. *Third*, we add more exploration studies on the different aspects of TSN framework and more experimental investigation on three new datasets (i.e., THUMOS, ActivityNet, and Kinetics). *Finally*, based on our TSN framework, we present an effective and efficient action recognition solution for ActivtyNet Large Scale Activity Challenge 2016, which ranks #1 in untrimmed video classification among 24 teams, and give a detailed analysis on different components of our method to highlight the important ingredients. The code of our method and learned models are publicly available to facilitate future research.[1]

2 Related Work

Action recognition has been studied extensively in recent years and readers can refer to [34], [35], [36] for good surveys. Here, we only cover the work related to our method.

2.1 Video Representation

For action recognition in videos, the visual representation plays a crucial role. We roughly categorize the related action recognition approaches into two types: methods based on *hand-crafted* features and those using *deeply-learned* features.

Hand-Crafted Features. In recent years, researchers have developed many different spatio-temporal feature detectors for video, such as 3D-Harris [37], 3D-Hessian [38], Cuboids [39], Dense Trajectories [40], Improved Trajectories [2]. Usually, a local 3D-region is extracted around the interest points or trajectories, and a histogram descriptor is computed to capture the appearance and motion information, such as Histogram of Gradient and Histogram of Flow (HOG/HOF) [41], Histogram of Motion Boundary (MBH) [40], 3D Histogram of Gradient (HOG3D) [42], Extended SURF (ESURF) [38], and so on. Then encoding methods are employed to aggregate these local descriptors into a global representation, and typical encoding methods include Bag of Visual Words (BoVW) [43], Fisher vector (FV) [44], Vector of Locally Aggregated Descriptors (VLAD) [45], and Multi-View Super Vector (MVSV) [46]. These local features share the merits of locality and simplicity, but may lack semantic and discriminative capacity.

To overcome the limitation of local descriptors, several mid-level representations have been proposed for action recognition [3], [23], [47], [48], [49], [50], [51]. Raptis et al. [47] grouped similar trajectories into clusters, each of which was regarded as an action part. Jain et al. [48] extended the idea of discriminative patches into videos and proposed discriminative spatio-temporal patches for representing videos. Zhang et al. [49] proposed to discover a set of mid-level patches in a strongly-supervised manner. Similar to 2-D poselet [52], they tightly clustered action parts using human joint labeling, dubbed *acteme*. Wang et al. [3] proposed a data-driven approach to discover those effective parts with high motion salience, known as *motionlet*. Zhu et al. [50] proposed a two-layer *acton* representation for action recognition. The weakly-supervised actons were learned via a max-margin multi-channel multiple instance learning framework. Wang et al. [23] proposed a multiple level representation called as *MoFAP* by concatenating motion features, atoms, and phrases. Sadanand et al. [51] presented a high-level video representation called as *Action Bank* by using a set action templates to describe the video content. In summary, these mid-level representations have the merits of representative and discriminative power, but still depends on the low-level hand-crafted features.

Deeply-Learned Features. Several works have been trying to learn deep features and design effective ConvNet architectures for action recognition in videos [1], [4], [5], [15], [16], [24], [25], [53], [54], [55]. Karpathy et al. [15] first tested ConvNets with deep structures on a large dataset (Sports-1M). Simonyan et al. [1] designed two-stream ConvNets containing spatial and temporal nets by exploiting ImageNet dataset for pre-training and calculating optical flow to explicitly capture motion information. Tran et al. [16] explored 3D ConvNets [53] on the realistic and large-scale video datasets, where they tried to learn spatio-temporal features with the operations of 3D convolution and pooling. Carreira et al. [56] proposed a new Two-Stream Inflated 3D CNNs (I3D) based on 2D CNN inflation, which allows for pre-training with ImageNet models. Sun et al. [54] proposed a factorized spatio-temporal ConvNets and exploited different ways to decompose 3D convolutional kernels, and Qiu et al. [57] designed a new Pseudo-3D Residual Networks by implementing spatio-temporal factorization with a residual learning module. Wang et al. [5] proposed a hybrid representation by using trajectory-pooled deep-convolutional descriptors (TDD), which share the merits of improved trajectories [2] and two-stream ConvNets [1]. Feichtenhofer et al. [58] further extended the two-stream ConvNets with convolutional fusion of two streams. Several works [4], [25], [55] tried to use recurrent neural networks (RNN), in particular LSTM, to model the temporal evolution of frame features for action recognition in videos.

Our work is related to those deep learning methods. In fact, any existing ConvNet architecture can work with the TSN framework, and thus be combined with the proposed sparse sampling strategy and aggregation functions to enhance the modeling capacity with long-range information. Meanwhile, our TSN is an end-to-end architecture, where the model parameters could be jointly optimized with the standard back propagation algorithm.

2.2 Temporal Structure Modeling

Many research works have been devoted to modeling the temporal structure of video for action recognition [19], [20], [21], [22], [59], [60]. Gaidon et al. [20] annotated each atomic action for each video and proposed Actom Sequence Model (ASM) for action detection. Niebles et al. [19] proposed to use latent variables to model the temporal decomposition of complex actions, and resorted to the Latent SVM [61] to learn the model parameters in an iterative approach. Wang et al. [21] and Pirsiavash et al. [59] extended the temporal decomposition of complex action into a hierarchical manner using Latent Hierarchical Model (LHM) and Segmental Grammar Model (SGM), respectively. Wang et al. [60]

1. https://github.com/yjxiong/temporal-segment-networks/

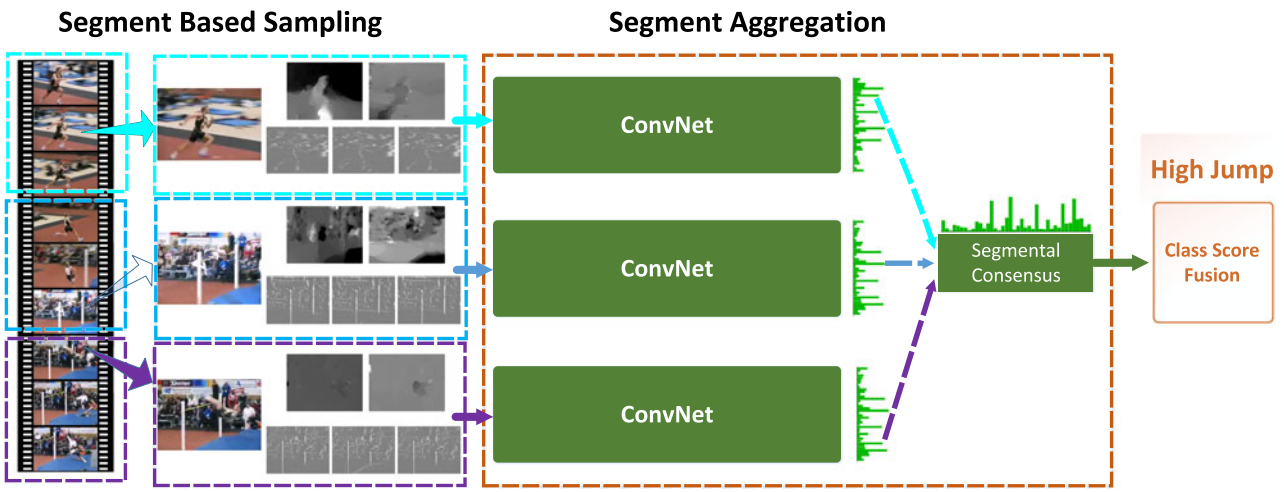


Fig. 1. *Temporal segment network*: One input video is divided into K segments (here we show the $K = 3$ case) and a short snippet is randomly selected from each segment. The snippets are represented by modalities such as RGB frames, optical flow (upper grayscale images), and RGB difference (lower grayscale images). The class scores of different snippets from the same modality are fused by a segmental consensus function to yield a video-level prediction. ConvNets on all snippets share the same parameters.

designed a sequential skeleton model (SSM) to capture the relation among dynamic-poselets, and performed spatio-temporal action detection. Fernando et al. [22] modeled the temporal evolution of BoVW representations for action recognition.

Several recent works focused on modeling long-range temporal structure with ConvNets [4], [24], [25], [58]. In general, these methods directly operated on a continuous video frame sequence with recurrent neural networks [4], [25], [55] or 3D ConvNets [24], [58]. Although these methods aim to deal with longer video duration, they usually process sequences of fixed length ranging from 5 to 120 frames due to the limit of computational cost and GPU memory. It is still non-trivial for these methods to learn from the entire video due to their limited temporal coverage. Our method differs from these end-to-end deep ConvNets by its novel adoption of a sparse temporal sampling strategy, which enables efficient learning using the entire video without using the entire videos without the limitation of sequence length. Therefore, our temporal segment network is a video-level and end-to-end framework for temporal structure modeling on the entire video.

3 Temporal Segment Networks

In this section, we give a detailed description of our temporal segment network framework. Specifically, we first discuss the motivation of segment based sampling. Then, we introduce the architecture of temporal segment network framework. After this, we present several aggregating functions of temporal segment network and provide analysis on these functions. Finally, we investigate some practical issues for the instantiation of temporal segment network framework.

3.1 Segment Based Sampling

As discussed in Section 1, long-range temporal modeling is important for action understanding in videos. The existing deep architectures such as two-stream ConvNets [1] and 3D ConvNets [16] are designed to operate on a single frame or a stack of frames (e.g., 16 frames) with limited temporal durations. Therefore, these architectures lack capacity of incorporating long-range temporal information of videos into the learning of action models.

In order to model long-range temporal structure, several approaches have been proposed to stack more consecutive frames at a fixed sampling rate [4], [24], [58]. Although this *dense* and *local* sampling could help to relieve the problem of the original short-term CovNets [1], [16], it still suffers in both *computational* and *modeling* aspects. From the computational perspective, it would greatly increase the cost of ConvNet training, as this dense sampling usually requires a large number of frames to capture long-range structure. For example, it totally samples 100 frames in the work of [24] and 120 frames for the method of [4]. From the modeling perspective, its temporal coverage is still local and limited by its fixed sampling interval, failing to capture the visual content over the entire video. For instance, the sampled 100 frames [24] only occupy a small portion of a 10-second video (around 300 frames).

We observe that *although the frames are densely recorded in the videos, the content changes relatively slowly*. This motivates us to explore a new paradigm for temporal structure modeling, called *segment based sampling*. This strategy is essentially a kind of *sparse* and *global* sampling method. Concerning the property of spareness, only a small number of sparsely sampled snippets would be used to model the temporal structure in a human action. Normally, the number of sampled snippets for one training iteration is fixed to a pre-defined value independent of the duration of a video. This guarantees that the computational cost will be constant, regardless of the temporal range we are dealing with. On the global property, our segment based sampling ensures these sampled snippets would distribute uniformly along the temporal dimension. Therefore, no matter how long an action video will last for, our sampled snippets would always roughly cover the visual content of the whole video, enabling us to model the long-range temporal structure throughout the entire video. Based on this paradigm for temporal structure modeling, we propose temporal segment network, a video-level training framework as shown in Fig. 1, which would be explained in the next subsection.

3.2 Framework and Formulation

We aim to design an effective and efficient video-level framework, coined Temporal Segment Network (TSN), by using a new strategy of segment based sampling. Instead of working on a single frame or a short frame stack, temporal segment networks operate on a sequence of short snippets sampled from the entire video. To make these sampled snippets represent the content of the whole video while still keeping reasonable computational cost, our segment based sampling first divides the video into several segments of equal duration, and then one snippet is randomly sampled from its corresponding segment. Each snippet in this sequence produces its own snippet-level prediction of the action classes, and a consensus function is designed to aggregate these snippet-level predictions into the video-level score. This video-level score is more reliable and informative than the original snippet-level prediction, since it captures the long-range information over the entire video. During the training process, the optimization objectives are defined on the video-level prediction and optimized by iteratively updating the model parameters.

Formally, given a video V, we divide it into K segments $\{S_1, S_2, \ldots, S_K\}$ of equal duration. One snippet T_k is randomly sampled from its corresponding segment S_k. Then, the TSN framework models a sequence of snippets $(T_1, T_2, \ldots, T_K)$ as follows:

$$\begin{aligned}&\mathrm{TSN}(T_1, T_2, \ldots, T_K)\\ &= \mathcal{H}(\mathcal{G}(\mathcal{F}(T_1; \mathbf{W}), \mathcal{F}(T_2; \mathbf{W}), \ldots, \mathcal{F}(T_K; \mathbf{W}))).\end{aligned} \tag{1}$$

Here, the temporal duration of each snippet T_k depends on the input modality and it could be 1 frame for RGB or 5 frames for Optical Flow and RGB Difference. $\mathcal{F}(T_k; \mathbf{W})$ is the function representing a ConvNet with parameters $\mathbf{W}$ which operates on the short snippet T_k and produces the scores over all the classes. The segmental consensus function $\mathcal{G}$ combines the outputs from multiple short snippets to obtain a consensus of class hypothesis among them. Based on this consensus, the function $\mathcal{H}$ predicts the probability of each action class for the whole video. Here we choose the widely used Softmax function for $\mathcal{H}$. In our TSN framework, the form of consensus function $\mathcal{G}$ is of great importance, as it should be equipped with high modeling capacity while still could be differentiable or at least has subgradients. The high modeling capacity refers to the ability to effectively aggregate snippet-level predictions into a video-level score and the differentiability allows our TSN framework to be easily optimized using backpropagation. We will provide the details on these consensus functions in the next subsection.

Combining standard categorical cross-entropy loss, the final loss function regarding the segmental consensus $\mathbf{G} = \mathcal{G}(\mathcal{F}(T_1; \mathbf{W}), \mathcal{F}(T_2; \mathbf{W}), \ldots, \mathcal{F}(T_K; \mathbf{W}))$ is formed as:

$$\mathcal{L}(y, \mathbf{G}) = -\sum_{i=1}^{C} y_i \left(g_i - \log \sum_{j=1}^{C} \exp g_j \right), \tag{2}$$

where C is the number of action classes, y_i is the groundtruth label concerning class i, and g_j is the jth dimension of $\mathbf{G}$. During the training phase of our TSN framework, the gradients of the loss value $\mathcal{L}$ with respect to model parameters $\mathbf{W}$ can be derived as

$$\frac{\partial \mathcal{L}(y, \mathbf{G})}{\partial \mathbf{W}} = \frac{\partial \mathcal{L}}{\partial \mathbf{G}} \sum_{k=1}^{K} \frac{\partial \mathbf{G}}{\partial \mathcal{F}(T_k)} \frac{\partial \mathcal{F}(T_k)}{\partial \mathbf{W}}, \tag{3}$$

where K is number of segments in TSN. When we use a gradient-based optimization method, such as stochastic gradient descent (SGD), to learn the model parameters, Eq. (3) shows that the parameter updates are utilizing the segmental consensus $\mathbf{G}$ derived from all snippet-level predictions. In this sense, TSN can learn model parameters from the entire video rather than a short snippet. Furthermore, by fixing K for all videos, we assemble a sparse temporal sampling to select a small number of snippets. It drastically reduces the computational cost for evaluating ConvNets on the frames, compared with previous works using densely sampled frames [4], [24], [25].

3.3 Aggregation Function and Analysis

As analyzed above, the consensus (aggregation) function is an important component in our TSN framework. In this section, we give a detailed description about the design of aggregation functions and derive their gradients with respect to snippet-level prediction scores. We also analyze the properties of different kinds of aggregation functions and provide some modeling insight. Specifically, we propose five types of aggregation functions: max pooling, average pooling, top-$\mathcal{K}$ pooling, weighted average, and attention weighting.

Max Pooling. In this aggregation function, we apply max pooling to the prediction score of each category among the sampled snippets, i.e., $g_i = \max_{k \in \{1,2,\ldots,K\}} f_i^k$, where f_i^k is the ith element of $\mathbf{F}^k = \mathcal{F}(T_k; \mathbf{W})$. The partial derivative of g_i with respect to f_i^k can be easily computed as

$$\frac{\partial g_i}{\partial f_i^k} = \begin{cases} 1, & \text{if } \ k = \mathrm{argmax}_l f_i^l, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

The basic idea of max pooling is to seek a *single* and most discriminative snippet for each action class and utilize this strongest activation as the video-level response of this category. Intuitively, it devotes its emphasis to a single snippet, while completely ignoring the responses of other snippets. Thus, this aggregating function encourages temporal segment network to learn from a most discriminative snippet for each action class, but lacks the capacity of jointly modeling multiple snippets for a video-level action understanding.

Average Pooling. One alternative to max pooling aggregation function is the average pooling, where we perform average operation over these snippet-level prediction scores for each class, i.e., $g_i = \frac{1}{K} \sum_{k=1}^{K} f_i^k$. The partial derivative of average aggregation function g_i withe respect to f_i^k is derived as follows:

$$\frac{\partial g_i}{\partial f_i^k} = \frac{1}{K}. \tag{5}$$

The basic intuition behind average pooling is to leverage the responses of *all* snippets for action recognition, and use their

mean activation as the video-level prediction. In this sense, average pooling is able to jointly model multiple snippets and capture the visual information from the whole video. On the other hand, in particular for noisy videos with complex background, some snippets may be action-irrelevant and averaging over these background snippets may hurt the final recognition performance.

Top-$\mathcal{K}$ Pooling. To strike a balance between max pooling and average pooling, we propose a new aggregation function, named *Top-$\mathcal{K}$ pooling*. In this aggregation function, we first select $\mathcal{K}$ most discriminative snippets for each action category and then perform average pooling over these selected snippets, i.e., $g_i = \frac{1}{\mathcal{K}} \sum_{k=1}^{\mathcal{K}} \alpha_k f_i^k$, where α_k is the indicator of selection, and is set as 1 if selected and otherwise 0. Max pooling and average pooling can be considered as special cases of top-$\mathcal{K}$ pooling, where $\mathcal{K}$ is set to 1 or K, respectively. Similarly, the partial derivative of g_i with respect to f_i^k can be computed as follows:

$$\frac{\partial g_i}{\partial f_i^k} = \begin{cases} \frac{1}{\mathcal{K}}, & \text{if } \alpha_k = 1, \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

Intuitively, this aggregation function is able to determine a subset of discriminative snippets adaptively for different videos. As a result, it shares merits of both max pooling and average pooling, having capacity of jointly modeling multiple relevant snippets while avoiding the influence of background snippets.

Linear Weighting. In this aggregation function, we aim to perform an element-wise weighted linear combination on the prediction score for each action category. Specifically, we define the aggregation function as $g_i = \sum_{k=1}^{K} \omega_k f_i^k$, where w_k is the weight for the kth snippet. In this aggregation function, we introduce a model parameter ω and compute the partial derivative of g_i with respect to f_i^k and w_k as follows:

$$\frac{\partial g_i}{\partial f_i^k} = \omega_k, \quad \frac{\partial g_i}{\partial \omega_k} = f_i^k. \tag{7}$$

In practice, we use this equation to update the network weights $\mathbf{W}$ and the combination weights ω alternatively. The basic assumption underlying this aggregation function is that action can be decomposed into several phases and these different phases may play different roles in recognizing action classes. This aggregation function is expected to learn importance weights of different phases of an action class. Compared with previous pooling based aggregation functions, this linear weighting acts as a soft version of snippet selection.

Attention Weighting. It is obvious that the above linear weighting scheme is data independent, thus lacking the capacity of considering the difference between videos. To overcome this limitation, we propose an adaptive weighting method, called *attention weighting*. In this aggregation function, we aim to learn a function to automatically assign an importance weight to each snippet according to the video content. Formally, the aggregation function is defined as $g_i = \sum_{k=1}^{K} \mathcal{A}(T_k) f_i^k$, where $\mathcal{A}(T_k)$ is the attention weight for snippet T_k and calculated according to video content adaptively. Within this formulation, we could calculate the partial derivative of g_i with respect to f_i^k and $\mathcal{A}(T_k)$ as follows:

$$\frac{\partial g_i}{\partial f_i^k} = \mathcal{A}(T_k), \quad \frac{\partial g_i}{\partial \mathcal{A}(T_k)} = f_i^k. \tag{8}$$

In this attention weighting scheme, the design of attention weighting function $\mathcal{A}(T_k)$ is crucial for final performance. In the current implementation, we first extract visual feature $\mathbf{R} = \mathcal{R}(T_k)$ from each snippet with the same ConvNet and then produce the attention weights as

$$\begin{aligned} e_k &= \omega^{att} \mathcal{R}(T_k), \\ \mathcal{A}(T_k) &= \frac{\exp(e_k)}{\sum_{l=1}^{K} \exp(e_l)}, \end{aligned} \tag{9}$$

where ω^{att} is the parameter of attention weighting function and will be learned jointly with network weights $\mathbf{W}$. Here $\mathcal{R}(T_k)$ is the visual feature for the kth snippet. Currently it is the activation of last hidden layer. Within this formation, we can calculate the partial derivative of $\mathcal{A}(T_k)$ with respect to attention model parameter ω^{att} as

$$\frac{\partial \mathcal{A}(T_k)}{\partial \omega^{att}} = \sum_{l=1}^{K} \frac{\partial \mathcal{A}(T_k)}{\partial e_l} \mathcal{R}(T_l), \tag{10}$$

where the partial derivative of $\frac{\partial \mathcal{A}(T_k)}{\partial e_l}$ is computed as

$$\frac{\partial \mathcal{A}(T_k)}{\partial e_l} = \begin{cases} \mathcal{A}(T_k)(1 - \mathcal{A}(T_l)), & \text{if } l = k, \\ -\mathcal{A}(T_k)\mathcal{A}(T_l), & \text{otherwise.} \end{cases} \tag{11}$$

Having this partial derivative formula, we can learn the attention model parameters ω^{att} using back-propagation together with the ConvNet parameters $\mathbf{W}$. In addition, due to the introduction of attention model $\mathcal{A}(T_k)$, the basic back-propagation formula in Eq. (3) should be rectified as follows:

$$\frac{\partial \mathcal{L}(y, \mathbf{G})}{\partial \mathbf{W}} = \frac{\partial \mathcal{L}}{\partial \mathbf{G}} \sum_{k=1}^{K} \left(\frac{\partial \mathbf{G}}{\partial \mathcal{F}(T_k)} \frac{\partial \mathcal{F}(T_k)}{\partial \mathbf{W}} + \frac{\partial \mathbf{G}}{\partial \mathcal{A}(T_k)} \frac{\partial \mathcal{A}(T_k)}{\partial \mathbf{W}} \right). \tag{12}$$

Overall, the advantages of introducing attention model $\mathcal{A}(T_k)$ come from two aspects: (1) The attention model enhances the modeling capacity of our temporal segment network framework by automatically estimating the importance weight of each snippet based on the video content. (2) Due to the fact that the attention model is based on ConvNet representations $\mathbf{R}$, it leverages extra backpropagation information to guide the learning process of ConvNet parameter $\mathbf{W}$ and may accelerate the convergence of training.

3.4 TSN in Practice

Temporal segment network provides a general framework to perform video-level learning. In order to train TSN models to achieve optimal performance, a few practical issues have to be taken into account. To this end, we study a series of practical matters from the aspects of TSN architectures, TSN inputs, and TSN training.

TSN Architectures. Our TSN is a general and flexible framework for video-level learning. To demonstrate the generality of our approach, we instantiate TSN with multiple network architectures. Specifically, for *ablation studies* on standard action recognition benchmarks, we choose the Inception architecture with Batch Normalization (BN-Inception) [62] due to its good balance between accuracy and efficiency. Compared with other ConvNet architectures

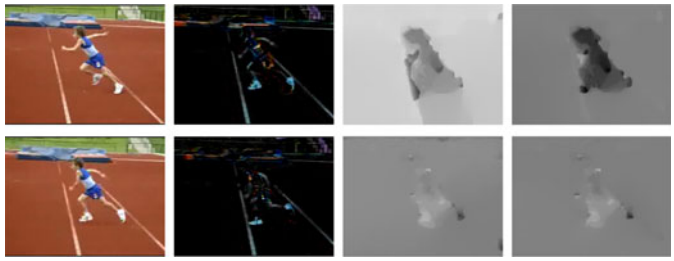


Fig. 2. Examples of four types of input modalities: RGB images, RGB difference, optical flow field (x,y directions), and warped optical flow field (x,y directions).

deployed in videos [1], [16], this architecture is equipped with better modeling capacity, allowing to demonstrate the improvement of TSN against a strong baseline. In the ActivityNet Challenge 2016, we investigate more powerful architectures including the Inception V3 [32] and ResNet-152 [31], to fully unleash the potential of TSN framework in video classification. In addition, we also instantiate the TSN with the architecture of I3D [56] on the Kinetics dataset, which unifies the short-term modeling (3D convolution and pooling) and long-term training.

TSN Inputs. Unlike static images, the additional temporal dimension of videos delivers another important cue for action understanding, namely motion. In [1], using dense optical flow field as the source of motion representation is proven to be effective. In this work, we extend this approach in two aspects, namely accuracy and speed. As shown in Fig. 2, in addition to the original input modalities of RGB and optical flow [1], we also investigate two other modalities: warped optical flow and RGB differences.

(1) *Warped Optical Flow.* Inspired by the work of improved dense trajectories [2], we investigate using warped optical flow field as the source for motion modeling. Warped optical flow field is known to be robust to camera motion and helps concentrate on human motion. We expect this to help to improve the accuracy in motion perception and thus boost the action recognition accuracy.

(2) *Stacked RGB Difference.* Despite the superior recognition accuracy, one issue that impedes the application of two-stream based approaches is the tremendous time cost of optical flow extraction. To address this problem, we build a motion representation without optical flow. Inspired by the success of frame volumes [16] and motion vector [17] in motion modeling, we revisit the simplest cue for apparent motion perception: the stacked difference of RGB pixel intensities between consecutive frames. Recalling the seminal work on dense optical flow in [63], the partial derivatives of pixel intensities with respect to time play critical roles in computing optical flow. It is reasonable to hypothesize that the power of optical flow in representing motion could be learned from the simple cues of RGB difference. This motivates us to investigate using RGB difference as the input of the temporal stream, which greatly saves the time of optical flow extraction.

TSN Training. As discussed before, existing human annotated datasets for action recognition are limited in terms of sizes. In practice, training deep ConvNets on these datasets is prone to over-fitting. To mitigate this issue, we design several strategies to improve the training in the TSN framework.

(1) *Cross Modality Initialization.* Pre-training the network parameters on large-scale image recognition datasets, such as ImageNet [18], has turned out to be an effective remedy when the target dataset does not have enough training samples [1]. As spatial networks take RGB images as inputs, it is natural to exploit models trained on the ImageNet as initialization. For other input modalities such as optical flow and RGB difference, we come up with a cross modality initialization strategy. Specifically, we first discretize optical flow field into the interval of 0 to 255 by linear transformation. Then, we average the weights of pretrained RGB models across the RGB channels in the first layer and replicate the mean by the channel number of temporal network input. Finally, the weights of remaining layers of the temporal network are directly copied from the pretrained RGB networks.

(2) *Regularization.* Batch Normalization [62] is able to deal with the problem of covariate shift by estimating the activation mean and variance within each batch to normalize these activation values. This operation speeds up the convergence of training, but also increases the risk of over-fitting in the transfer learning process, due to the biased estimation of mean and variance from a limited number of training samples in target dataset. Therefore, after initialization with pre-trained models, we choose to freeze the mean and variance parameters of all Batch Normalization layers except the first one. As the distribution of optical flow is different from the RGB images, the activation value of first convolution layer will have a distinct distribution and we need to re-estimate the mean and variance accordingly. We call this strategy *partial BN*. Meanwhile, we add an extra *dropout* layer with high dropout ratio (set as 0.8 in experiment) after the global pooling layer to further reduce the effect of over-fitting on the small datasets such as UCF101 and HMDB51.

(3) *Data Augmentation.* In the original two-stream ConvNets [1], random cropping and horizontal flipping are employed to augment training samples. We exploit two new data augmentation techniques: corner cropping and scale-jittering. In the corner cropping technique, the extracted regions are only selected from the corners or the center of an image to avoid implicitly focusing more on the center area. In the multi-scale cropping technique, we adapt the scale jittering technique [8] used in ImageNet classification to action recognition. We present an efficient implementation of scale jittering. We fix the input size as 256×340, and the width and height of cropped regions are randomly selected from $\{256, 224, 192, 168\}$. Finally, these cropped regions will be resized to 224×224 for network training. In fact, this implementation not only contains scale jittering, but also involves aspect ratio jittering.

4 Action Recognition with TSN Models

With the principled framework of temporal segment network, there still remains the question of how to use the models

learned with this framework to recognize actions in realistic videos. In this section, we describe how to apply action models under two different conditions: trimmed videos and untrimmed videos, and devise a series of techniques to improve the robustness of action recognition.

4.1 Action Recognition in Trimmed Video

In trimmed videos, action instances are manually cropped from the long video sequences and thereby action recognition could be simply cast as a classification problem. Due to the fact that all snippet-level ConvNets share the model parameters in temporal segment networks, the learned models can perform snippet-wise evaluation as normal ConvNets [1]. This also allows us to carry out fair comparison with models learned without the TSN framework. Specifically, we follow the testing scheme of the original two-stream ConvNets [1], where we sample 25 snippets of different modalities. Meanwhile, we crop 4 corners and 1 center, and their horizontal flipping from the sampled snippets to evaluate the ConvNets. We use average pooling to aggregate the predictions of different crops and snippets. For the fusion of predictions from multiple modalities, we take a weighted average of them, where the fusion weights are determined empirically. It is described in Section 3.2 that the segmental consensus function is applied before the Softmax normalization. To test the models in compliance with their training, we fuse the prediction scores of 25 frames and different streams before Softmax normalization.

4.2 Action Recognition in Untrimmed Videos

The major obstacle for action recognition in untrimmed videos is the large portion of irrelevant content in the input videos. Since our action models are trained on trimmed action clips, reusing the technique used for trimmed video, i.e., simply averaging scores from every location in a video, has a high risk of factoring in the unpredictable responses of the models on background content. This makes it necessary to design a specialized method for applying the trained action recognition models to untrimmed videos. For this purpose, we start by summarizing the following challenges posed by untrimmed videos.

- Location issue: an action clip can appear at any temporal location of the video.
- Duration issue: the action clip can be either long-lasting or ephemeral.
- Background issue: the irrelevant content in a video can have high variations and can possibly occupy a large portion of the whole duration of a video.

To deal with these challenges, we develop a detection based method to apply action models to untrimmed videos. First, to cover any location that the action instance can reside, we sample snippets from the input videos in a fixed sampling rate (e.g., 1FPS). A trained TSN model is then evaluated on these sampled snippets. Then, in order to cover the highly varying durations of action clips, a series of temporal sliding windows with different sizes are then applied on the snippet scores. The maximum score of the classes within a window is used to represent it. To alleviate the interference of background content, windows with the same length are then aggregated with a top-$\mathcal{K}$ pooling scheme. The aggregation result from different window sizes then votes for the final prediction of the whole video.

Formally, for a video in length of M seconds, we will obtain M snippets $\{T_1, \ldots, T_M\}$. Applying the TSN model, we will obtain class scores $\mathcal{F}(T_m)$ for the snippet T_m. We then build temporal sliding windows with the size of $l \in \{1, 2, 4, 8, 16\}$. The windows will slide through the whole duration of videos, with a stride of $0.8 \times l$. For a window position starting at the sth second, a series of snippets will be covered as $\{T_{s+1}, \ldots, T_{s+l}\}$, with their class scores $\{\mathcal{F}(T_{s+1}), \ldots, \mathcal{F}(T_{s+l})\}$. The class scores for this window $\mathbf{F}^{s,l}$ can be calculated by

$$F_i^{s,l} = \max_{p \in \{1,2\ldots,l\}} \{f_i^{s+p}\}. \tag{13}$$

In this way, for size l we will obtain N^l windows, where $N^l = \lfloor \frac{M}{0.8l} \rfloor$. Then we apply the aforementioned top-$\mathcal{K}$ pooling scheme to obtain the consensus $\mathbf{G}^l$ of from these N^l windows of size l. Here the parameter $\mathcal{K}$ is determined as $\mathcal{K} = \max(15, \lceil N^l/4 \rceil)$. This gives us 5 sets of class scores for window size $l \in \{1, 2, 4, 8, 16\}$. The final score is then calculated as $\mathbf{P} = \frac{1}{5} \sum_{l \in \{1,2,4,8,16\}} \mathbf{G}^l$, which is the average of the five window sizes. We term this video classification technique as *Multi-scale Temporal Window Integration*, abbreviated as M-TWI.

5 Experiments

In this section, we first introduce the evaluation datasets and implementation details of our approach. Then we discuss the practical issues for action recognition with deep learning and our proposed good practices to mitigate them. After dealing with these issues, we provide detailed analysis of the proposed temporal segment network framework, to demonstrate the importance of modeling long-term temporal structure. Finally, we compare the performance of our method with the state of the art on the five action recognition benchmarks. We also present the results of our approach in the ActivityNet challenge 2016 and describe the winner solution to this challenge. Additionally, we visualize our learned ConvNet models to help qualitatively justify the performance improvement.

5.1 Datasets

The datasets adopted to evaluate the performance of temporal segment network framework are from two types of videos, i.e., trimmed videos and untrimmed videos. Now we describe the details of these datasets.

Trimmed Video Datasets. We conduct experiments on three standard action recognition datasets of trimmed videos, namely HMDB51 [29], UCF101 [28], and Kinetics 400 [30]. The UCF101 dataset contains 101 action classes and 13,320 video clips. We follow the evaluation scheme of the THUMOS13 challenge [64] and adopt the three training/testing splits for evaluation. The HMDB51 dataset is a large collection of realistic videos from various sources, such as movies and web videos. The dataset is composed of 6,766 video clips from 51 action categories. Our experiments follow the original evaluation scheme using three training/testing splits and report *average accuracy* over these splits. The Kinetics400 dataset is the largest well-labeled action recognition dataset. Its current version contains 400 action

TABLE 1
Exploration of Different Training Strategies for Two-Stream ConvNets on the UCF101 Dataset (*Split 1*)

Training setting	Spatial	Temporal	Two-Stream
Baseline [1]	72.7%	81.0%	87.0%
From Scratch	48.7%	81.7%	82.9%
Pre-train Spatial	84.1%	81.7%	90.0%
+ Cross modality pre-training	84.1%	86.6%	91.5%
+ Partial BN with dropout	84.5%	87.2%	92.0%
without corner cropping	84.2%	86.8%	91.8%

Here, "from scratch" refers to the case we initialize the ConvNet parameters with Gaussian distribution. "pre-train spatial" means only pre-training spatial stream ConvNet while training temporal stream ConvNet from scratch. Experiments here are conducted without *TSN.*

TABLE 2
Exploration of Different Combination of Modalities with TSN on the UCF101 Dataset (*Over Three Splits*)

Modalities	TSN	Accuracy	Speed (FPS)
RGB+Flow	No	92.4%	14
RGB+Flow	Yes	94.9%	14
RGB+Flow+Warp	Yes	95.0%	5
EMV [17]	-	81.6%[a]	480
RGB Diff.	No	84.2%	660
RGB Diff.	Yes	87.7%	660
Two-Stream 3D ConvNet [68]	-	90.2%	246
RGB+EMV [17]	-	86.4%	390
RGB+RGB Diff.	No	86.8%	340
RGB+RGB Diff.	Yes	91.0%	340

In this table, "RGB" refers to the RGB video frame stream. "Flow" refers to modality of optical flow input. "Warp" refers to the modality of warped optical flow. "RGB Diff." refers to the modality using difference of RGB video frames. The speed for testing is evaluated on a TitanX GPU. In the lower half of the table, we compare "RGB+RGB Diff." with other real-time action recognition methods (FPS $>$ 30).
[a]This result is got by personal communication with the first author of [17].

classes and each category has at least 400 videos. In total, there are around 240,000 training videos, 20,000 validation videos, and $40,000$ testing videos. The evaluation metric on the Kinetics dataset is the top-1 and top-5 accuracy.

Untrimmed Video Datasets. We conduct experiments of untrimmed video action recognition on two publicly available large-scale datasets. The first is the THUMOS14 [65]. It has 101 classes of human actions. This dataset is composed of training set, validation set, testing set, and background set. We use the training set (UCF101) and validation set ($1,010$ videos) for TSN training and evaluate the learned models on its testing set, which has $1,575$ videos. The second dataset for untrimmed videos is the ActivityNet [27] dataset. We use its release version 1.2, termed as ActivityNet v1.2. The ActivityNet v1.2 dataset has 100 classes of human activities. It consists of $4,819$ videos for training, $2,383$ videos for validation, and $2,480$ videos for testing. We follow the standard splits to train and evaluate the TSN framework. On both datasets of untrimmed videos, the evaluation metric is *mean average precision (mAP)* for action recognition.

5.2 Implementation Details

We use the mini-batch stochastic gradient descent algorithm to learn the network parameters, where the batch size is set to 128 and momentum set to 0.9. We initialize network weights with pre-trained models from ImageNet [18]. We set a smaller learning rate in our experiments. On the dataset of UCF101, for spatial networks, the learning rate is initialized as 0.001 and decreases to its $\frac{1}{10}$ every $1,500$ iterations. The whole training procedure stops at $3,500$ iterations. For temporal networks, we initialize the learning rate as 0.005, which reduces to its $\frac{1}{10}$ after $12,000$ and 18,000 iterations. The maximum number of iterations is set as 20,000. To speed up training, we employ a data-parallel strategy with multiple GPUs, implemented with our modified version of Caffe [66] and OpenMPI.[2] The whole training time on UCF101 is around 0.6 hours for spatial TSNs and 8 hours for temporal TSNs with 8 TITANX GPUs. For other datasets such as HMDB51, THUMOS14, ActivityNet, and Kinetics, the learning process is the same with that of UCF101, except that the iteration numbers are adjusted according to the dataset sizes. Concerning data augmentation, we use the techniques of location jittering, horizontal flipping, corner cropping, and scale jittering, as specified in Section 3.4. For the extraction of optical flow and warped optical flow, we choose the TVL1 optical flow algorithm [67] implemented by OpenCV with CUDA. *If not specifically noted, the experiments in the section are conducted with BN-Inception* [62] *as the underlying ConvNet architecture.*

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

5.3 Effectiveness of the Proposed Practices

In this section, we focus on investigating the effect of the good practices described in Section 3.4, including the training strategies and the input modalities. In this exploration study, we use the two-stream ConvNets with very deep architecture adapted from [62].

Different Learning Strategies. Compared with the original two-stream ConvNets [1], we propose two new training strategies in Section 3.4, namely cross modality pre-training and partial BN with dropout. Specifically, we compare four settings on the split1 of UCF101: (1) training from scratch; (2) only pre-train spatial stream as in [1]; (3) with cross modality pre-training; (4) combination of cross modality pre-training and partial BN with dropout. The results are summarized in Table 1. First, we see that the performance of training from scratch is much worse than that of the original two-stream ConvNets (baseline), which implies carefully designed learning strategy is necessary to reduce the risk of over-fitting, especially for spatial networks. Then, we resort to the pre-training of the spatial stream and cross modality pre-training of the temporal stream to help initialize two-stream ConvNets and it achieves better performance than the baseline. We further utilize the partial BN with dropout to regularize the training procedure, which boosts the recognition performance to 92.0 percent. We also perform a comparative experiment to verify the effectiveness of corner cropping by cropping regions in a 3×3 grid. We report the performance in the last row of Table 1 and its result is slightly worse than corner cropping. Therefore, in the remaining experiments, we employ all these good practices for model training.

Different Input Modalities. We propose two new types of modalities in Section 3.4: RGB difference and warped optical flow field. We try to combine different modalities and report the results in Table 2. These experiments are carried

TABLE 3
Comparison of Segment Based Sampling of TSN with the Regular Sampling on the UCF101 Dataset (*Over Three Splits*)

Sampling strategy	Spatial	Temporal	Two-Stream
Regular sampling (stride 1)	84.3%	85.6%	92.1%
Regular sampling (stride 5)	84.9%	86.7%	93.2%
Regular sampling (stride 10)	85.3%	86.6%	93.5%
Regular sampling (stride 15)	85.3%	86.5%	93.6%
Regular sampling (stride 20)	85.3%	86.6%	93.7%
Segment based sampling (TSN)	86.5%	89.8%	**94.2%**

For fair comparison, in addition to sampling strategy, the remaining implementation details are kept the same. The number of sampled snippets is set as 3.

out with all the good practices verified in Table 1. We perform multiple experiments with or without TSN (7 segments) to investigate the performance of different input modalities. We first observe that RGB and optical flow, which is the basic combination in the two-stream ConvNets, also works well with TSN, yielding recognition accuracy of 94.9 percent. Then we observe that the warped optical flow slightly increases the performance (94.9 to 95.0 percent), but severely slows down the testing speed to only 5 FPS. So we only use RGB and optical flow to report the final performance. Another interesting finding is that the simple motion representation of RGB difference, when used together with RGB data under TSN framework, can provide competitive recognition performance (91.0 percent) while running at a very fast speed of 340FPS. It also outperforms other state-of-the-art real-time action recognition methods as shown in Table 2. This suggests that "RGB + RGB Diff." can serve well for building real-time action recognition systems with moderate accuracy requirement.

5.4 Study on Temporal Segment Networks

In this section, we focus on studying the effectiveness of the TSN framework. As described in Section 3, the TSN framework is based on segment based sampling, which provides an efficient and effective scheme for video-level learning. We first compare segment based sampling with the regular sampling method [4], [24], [58] on the dataset of UCF101, to demonstrate the effectiveness of segment based sampling. In addition, the TSN framework has two other critical components: the sparse snippet sampling scheme and the segment consensus (aggregation) functions. To analyze the TSN framework in-depth, we first explore the effect of segment number and then analyze different consensus (aggregation) functions. These experiments are performed on the datasets of UCF101 and ActivityNet, to reflect the scenarios of both trimmed and untrimmed video action recognition. Finally, to further demonstrate the importance of TSN in long-range modeling, we also compare the performance of TSN with other very deep architectures on the UCF101 dataset.

Study on Segment Based Sampling . Our proposed segment based sampling is a global and sparse sampling method. Compared with those regular sampling methods [4], [24], [58], it aims to model long-term temporal structure in a more efficient and effective way. Here we perform an experimental comparison with the regular sampling method. Specifically, for regular sampling, we select T snippets (each snippet has a single frame for RGB and 5 stacked frames for Flow) at a fixed sampling stride τ. For fair comparison, the remaining implementation details are kept the same, such as sampled snippet number, training strategy, and snippet-level score fusion.

The experimental results are summarized in Table 3. In this experiment, we keep the number of sampled snippets as 3 and vary the sampling stride τ from 1 to 20. We see that increasing sampling stride would improve the performance of regular sampling (from 92.1 to 93.7 percent), as a larger sampling stride contributes to longer-term modeling. However, the performance of regular sampling is still lower than that of segment based sampling (93.7 percent versus 94.2 percent). We analyze that our proposed segment based sampling is adaptive to the video duration and capable of describing the visual content of entire video.

Evaluation on Segment Number. The most crucial parameter governing the segment based sampling scheme in TSN is the number of segments K. When K equals to 1, TSN degenerates to the plain two-stream ConvNets. Increasing K is expected to improve the recognition performance of the learned models. In experiments, we vary the number of K from 1 to 9 and evaluate the recognition performance using the same test approach.

The results are summarized in Table 4. We observe that increasing the segment number will generally lead to better performance. For example, the performance of TSN with 7 segments is better than that of TSN with 3 segments (94.9 percent versus 94.2 percent). This improvement implies that using more segments will help to capture richer information and better model temporal structure of the whole video. However, when the segment number K increases from 5 to 9, it brings a very small improvement. Thus, to strike a balance between recognition performance and computational burden, we set $K = 7$ in the following experiments.

TABLE 4
Exploration of Different Segment Numbers K in TSN on the UCF101 Dataset (*Over Three Splits*) and the ActivityNet Dataset (Train on the Training Set and Test on the Validation Set)

Dataset	UCF101			ActivityNet v1.2 Val.		
K	Spatial	Temporal	Two-Stream (1:1)	Spatial	Temporal	Two-Stream (1:0.5)
1	85.0%	88.3%	92.4%	82.0%	61.4%	84.7%
3	86.5%	89.8%	94.2%	83.6%	70.6%	86.9%
5	86.7%	90.1%	94.7%	84.6%	72.9%	87.6%
7	86.4%	90.1%	**94.9%**	84.0%	72.8%	87.8%
9	86.2%	89.7%	**94.9%**	83.7%	72.6%	**87.9%**

We use the average consensus function in these experiments.

TABLE 5
Exploration of Different Segmental Consensus Functions for TSN on the UCF101 Dataset (*Over Three Splits*) and the ActivtyNet Dataset (Train on the Training Set and Test on the Validation Set)

Dataset	UCF101			ActivityNet v1.2 Val.		
Consensus Function	Spatial	Temporal	Two-Stream (1:1)	Spatial	Temporal	Two-Stream (1:0.5)
Max Pooling	84.9%	83.5%	92.4%	81.8%	62.0%	85.4%
Average Pooling	86.4%	90.1%	**94.9%**	84.0%	72.8%	87.8%
Weighted Average	86.4%	89.7%	94.8%	83.1%	70.5%	86.4%
Top-$\mathcal{K}$ Pooling	85.5%	88.8%	94.2%	84.7%	73.6%	88.1%
Attention Weighting	86.1%	89.1%	94.6%	84.1%	71.8%	**88.2**%

We set segment number K as 7 in these experiments.

Evaluation on Aggregation Functions. In Eq. (1), a segmental consensus function is defined by its aggregation function $\mathcal{G}$, which could be crucial to the final recognition performance. Here we evaluate five candidates, including the relatively basic: (1) max pooling, (2) average pooling, (3) weighted average, and the more complex: (4) top-$\mathcal{K}$ pooling and (5) attention weighting, for the form of $\mathcal{G}$.

The experimental results are summarized in Table 5. On UCF101, which consists of trimmed human action videos, the average aggregation function achieves the best performance, and the weight average and attention weighting obtain quite similar performance. On ActivityNet, the top-$\mathcal{K}$ and attention weighting aggregation functions achieve the best performance, which slightly outperforms (0.4 percent) the basic ones such as average pooling. This fact suggests that on datasets with more complex and diverse temporal structure, the advanced aggregation functions will lead to better recognition accuracies. Therefore, we default to average pooling for short videos (HMDB51 and UCF101) and top-$\mathcal{K}$ pooling for complex videos (ActivityNet) in later experiments.

Comparison of ConvNet Architectures. We have conducted previous experiments mostly with the BN-Inception architecture. Here we compare the performance of different network architectures on the UCF101 dataset and the results are summarized in Table 6. We use $K = 1$ in these experiments, which is equivalent to the original two-stream ConvNets. Specifically, we compare the performance of four very deep architectures: BN-Inception [62], GoogLeNet [9], VGGNet-16 [8], and ResNet-152 [31]. The results of different architectures are directly cited from the corresponding references. Among the compared architectures, the very deep two-stream ConvNets adapted from BN-Inception [62] achieves the best accuracy of 92.4 percent, which is still better than the ResNet-152 by 0.6 percent. This performance improvement may be ascribed to the good practices proposed by our approach. Furthermore, when trained with TSN ($K = 7$), the accuracy is boosted to 94.9 percent. This clearly justifies the effectiveness of modeling long-range temporal structure with TSN.

TABLE 6
Comparison of Different ConvNet Architectures on the UCF101 Dataset (*Over Three Splits*)

Training setting	Spatial	Temporal	Two-Stream
VGG-M [69] (from [1])	73.0%	83.7%	86.9%
GoogLeNet [9] (from [70])	75.3%	85.8%	89.3%
VGGNet-16 [8] (from [70])	78.4%	87.0%	91.4%
ResNet-152 [31] (from [71])	83.4%	87.2%	91.8%
BN-Inception [62]	85.0%	88.3%	92.4%
BN-Inception+TSN	86.4%	90.1%	**94.9%**

"BN-Inception+TSN" refers to the setting where the TSN framework is applied on top of the best performing BN-Inception [62] architecture. It is worth noting that the reported BN-Inception result is not directly comparable to those previous works as its training is based on our proposed good training practices.

5.5 Comparison with the State of the Art

After analyzing the effect of the components in the TSN framework and coming to an optimal setting, we now compare our action recognition approach against the state-of-the-art methods on both trimmed videos and untrimmed videos. We conduct experiments on five action recognition datasets. The first three, HMDB51 UCF101, and Kinetics, are composed of trimmed videos. The last two, THUMOS14 and ActivityNet v1.2, consist of untrimmed videos. We expect the experimental results on these datasets would provide a thorough comparison with the existing state-of-the-art methods. In experiments, we use the RGB and optical flow modalities to make fair comparison with previous methods.

Trimmed Video Datasets. We experiment on three challenging trimmed video datasets: HMDB51 UCF101, and Kinetics. The results are summarized in Table 9, where we compare our method with both traditional approaches such as improved dense trajectories (iDTs) [2], MoFAP representations [23], and deep learning representations, such as 3D convolutional networks (C3D) [16], trajectory-pooled deep-convolutional descriptors (TDD) [5], factorized spatio-temporal convolutional networks ($\mathrm{F_{ST}CN}$) [54], long term convolution networks (LTC) [24], and key volume mining framework (KVMF) [78]. We present the results of TSN of 3 and 7 segments with the average aggregation function. We fuse the prediction scores of RGB and optical flow modalities with equal weights (1:1). Our best results outperform other methods by 5.5 percent on the HMDB51 dataset, and 1.8 percent on the UCF101 dataset. The superior performance of our method demonstrates the effectiveness of temporal segment network on trimmed videos and the importance of effective long-term temporal modeling. We also report the performance on the recent Kinetics dataset and build the TSN with I3D architecture and 2 segments, which combines the merits of short-term and long-term modeling. We see that against the very competitive I3D method, our TSN training is still able to improve the performance from 74.9 to 75.7 percent.

TABLE 7
Evaluation on the Validation Set of ActivityNet Challenge 2016 Data (ActivityNet v1.3 Val.)

Settings	mAP on ActivityNet v1.3 Val.		
	Spatial	Temporal	Two Stream
BN-Inception w/o TSN	76.6%	52.7%	78.9%
TSN + BN-Inception	79.7%	63.6%	84.7%
TSN + Inception V3	83.3%	64.4%	87.7%
TSN-Top3 + Inception V3	84.5%	64.0%	88.0%
TSN-Ensemble	85.9%	68.3%	**89.7%**

"BN-Inception w/o TSN" indicates that we train the models without TSN. "TSN+X", indicates that we train TSN with the underlying ConvNet architecture "X". "TSN-Top3" refers to the case where the top-$\mathcal{K}$ aggregation function is used with $\mathcal{K}$ set to 3. Here we use the fusion weights of 1:0.5 for RGB and optical flow, respectively.

Untrimmed Video Datasets. We also compare our approach with other methods on two untrimmed video datasets: THUMOS14 and ActivityNet v1.2. The results are summarized in Table 9. We compare TSN with the existing methods for untrimmed video action recognition, including improved dense trajectories [2], two-stream ConvNet [1], enhanced motion vectors [17], 3D convolutional networks [16], object+motion [73], and Depth2Action [74]. We also present the results of TSN with segment numbers of 3 and 7 and the aggregation function in TSN is top-$\mathcal{K}$ pooling. Our approach clearly outperforms these compared methods. For example, our TSN (7 seg) is better than the previous best performance by 8.5 percent on the THUMOS14 dataset and 11.5 percent on the ActivityNet dataset. This confirms that models learned with TSN also perform well in untrimmed videos, given a reasonable testing scheme, as described in Section 4.2.

5.6 ActivityNet Challenge 2016

The power of the TSN framework is further verified in the ActivityNet large scale activity recognition challenge 2016. In this challenge we use the videos from the ActivityNet [27] version 1.3 for training and testing. In Particular, we train TSN models using the trimmed activity instances from the ActivityNet v1.3. To test the models, we follow the approach described in Section 4.2. Understanding that the underlying ConvNet architecture plays an important role in boosting the performance, we also instantiate TSN with the ultra-deep Inception V3 [32] and ResNet-152 [31] architectures.

To evaluate the performance of TSN, we experiment with two settings. First we train the models on the "training" subset of ActivityNet v1.3 and test the recognition accuracy in terms of mean average precision on the "validation" subset. In the second setting, we train the models with both "training" and "validation" subsets and test the recognition accuracy on the "testing" subset. The mAP values on the "testing" subset are reported by the publicly available test server of the challenge.[3] The results on validation subset are summarized in Table 7. We observe that TSN significantly boosts the performance over plain two-stream ConvNets (from 78.9 to 84.7 percent). The performance gain is further amplified by using deep CNN architectures such as Inception V3. Also, the advanced aggregation function such as Top-$\mathcal{K}$ pooling leads to even better performance. Finally, we find that models trained with different aggregation functions (i.e., average pooling, Top-$\mathcal{K}$ pooling, attention weighting) and CNN architectures (i.e., Inception V3, ResNet-152) are complementary when combining into an ensemble, leading to an mAP value of 89.7 percent.

3. http://activity-net.org/challenges/2016/evaluation.html

TABLE 8
Winning Entries in the Untrimmed Video Classification Task of ActivityNet Challenge 2016 (ActivityNet v1.3 Test)

Team	mAP	Top1 Accuracy
CES (ours)	**93.23%**	**88.14%**
QCIS	92.41%	87.79%
MSRA	91.94%	86.69%
UTS	87.16%	84.90%
Tokyo Univ.	86.46%	80.43%

We present the recognition results in the form of mAP values and top-1 accuracies. The entries are ranked by mAP values in the challenge.

Challenge Solution and Result. The results on the testing set are summarized in Table 8. Out entry "CES" ranks first among all 24 challenge participants with an mAP of 93.23 percent on the testing set. The submission is an ensemble of TSN models trained on training and validation data with Inception V3 and ResNet-152 architectures, and the audio models [80] trained on audio signal of the videos. For references we also list the top results from other participants of this challenge in Table 8. It is worth noting that thanks to the high efficiency of TSN, our models in the challenge can be trained within 10 hours on a single node with 8 TitanX GPUs.

5.7 Model Visualization

Besides recognition accuracies, we would like to attain further insight into the learned ConvNet models. In this sense, we adopt the DeepDraw [79] toolbox. This tool conducts iterative gradient ascent on input images with only white noises. Thus the output after a number of iterations can be considered as class visualization solely based on class knowledge inside the ConvNet model. The original version of the tool only deals with RGB data. To conduct visualization on optical flow based models, we adapt the tool to work with our temporal ConvNets. As a result, we for the first time visualize interesting class information in two-stream ConvNet models. We randomly pick five classes from the UCF101 dataset, *Taichi*, *Punch*, *Diving*, *Long Jump*, and *Biking*, and five classes from the ActivityNet dataset, *Poole Vault*, *Zumba*, *Skate Board*, *Rock Climb*, and *Cheer Leading*. The results are shown in Fig. 3. For both RGB and optical flow, we visualize the ConvNet models learned with following two settings: (1) training without temporal segment network and (2) training with temporal segment network.

It is easy to notice that the models, trained with only short-term information such as a single frame, tend to focus more on the scenery patterns and objects in the videos. For example, in the class "Diving", the single-frame spatial stream ConvNet pays much attention on platforms while the person performing diving is less clear than these platforms. With the training of TSN, the spatial ConvNet is able to generate an image that human is the major visual

TABLE 9
Comparison of Our Method Based on TSN with Other State-of-the-Art Approaches on the Datasets of HMDB51, UCF101, THUMOS14, ActivityNet v1.2, and Kinetics400

HMDB51		UCF101		THUMOS14		ActivityNet v1.2 Test		Kinetics400 Val. (top1/top5)	
iDT+FV [2]	57.2%	iDT+FV [72]	85.9%	iDT+FV [72]	63.1%	iDT+FV [72]	66.5%	RGB-I3D [56]	72.9%/90.8%
DT+MVSV [46]	55.9%	DT+MVSV [46]	83.5%	object+motion [73]	71.6%	Depth2Action [74]	78.1%	RGB-I3D+TSN	73.5%/91.6%
iDT+HSV [75]	61.1%	iDT+HSV [75]	87.9%					FLOW-I3D [56]	65.3%/86.7%
MoFAP [23]	61.7%	MoFAP [23]	88.3%					FLOW-I3D+TSN	65.4%/86.7%
Two Stream ConvNet [1]	59.4%	Two Stream ConvNet [1]	88.0%	Two Stream ConvNet [1]	66.1%	Two Stream ConvNet [1]	71.9%	Two Stream ConvNet [1]	61.0%/81.3%
VideoDarwin [22]	63.7%	C3D (3 nets) [16]	85.2%	EMV+RGB [17]	61.5%	C3D [16]	74.1%	Two Stream ARTNet [76]	71.3%/89.5%
MPR [77]	65.5%	Two stream +LSTM [4]	88.6%					Two Stream I3D [56]	74.9%/91.8%
$F_{ST}CN$[54]	59.1%	$F_{ST}CN$ [54]	88.1%						
TDD+FV [5]	63.2%	TDD+FV [5]	90.3%						
LTC [24]	64.8%	LTC [24]	91.7%						
KVMF [78]	63.3%	KVMF [78]	93.1%						
TSN (3 seg)	70.7%	TSN (3 seg)	94.2%	TSN (3 seg)	78.8%	TSN (3 seg)	89.0%	TSN (2D ConvNet)	73.9%/91.1%
TSN (7 seg)	**71.0%**	TSN (7 seg)	**94.9%**	TSN (7 seg)	**80.1%**	TSN (7 seg)	**89.6%**	TSN (I3D)	**75.7%/92.5%**

First, we instantiate TSN with the two-stream 2D CovnNets (BN-Inception) on the five datasets. In addition, we also use two-stream I3D as the backbone architecture for TSN training on the Kinetics400 dataset. Note that the results of I3D on the validation set of Kinetics400 are reproduced by ourselves.

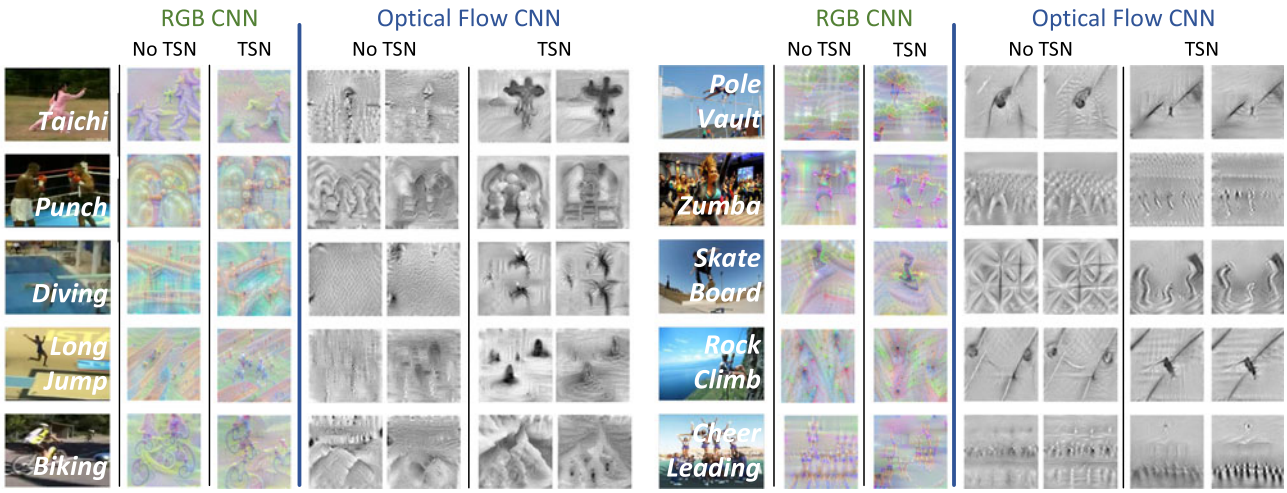


Fig. 3. Visualization of ConvNet models for action recognition using DeepDraw [79]. It is worth noting that video frames are just representatives of the corresponding classes, but not used for these visualizations. All these images are generated from purely random pixels. We compare two settings: (1) without temporal segment network (No TSN); (2) with temporal segment network (TSN). For spatial ConvNets, we plot two generated visualization as color images. For temporal ConvNets, we plot the flow maps of x (left) and y (right) directions in gray scales. *Left*: classes in UCF101. *Right*: classes in ActivityNet v1.2.

information and different poses can be identified. Its temporal stream counterpart, working on optical flow without TSN, tends to focus on the noisy motion pattern. With long-term temporal modeling of TSN, the learned models focus more on humans in the videos, and model the long-range structure of the action class. Similar observation would be identified in other action classes such as "Long Jump" and "Cheer Leading". This suggests that models learned with the proposed method may perform better, which is well reflected in our quantitative experiments.

6 Conclusions

In this paper, we have presented the Temporal Segment Network, a video-level framework that aims to model long-range temporal structure. As demonstrated on five action recognition benchmarks and ActivtyNet challenge 2016, this work has brought the state of the art to a new level, while maintaining a reasonable computational cost. This is largely ascribed to the segmental architecture with sparse sampling, as well as a series of good practices that we explored in this work. The former provides an effective and efficient way to capture long-range temporal structure, while the latter makes it possible to train very deep networks on a limited training set without severe over-fitting.

Acknowledgments

This work is supported by the National Science Foundation of China (No. 61321491, No. U1613211), Collaborative Innovation Center of Novel Software Technology and Industrialization, Shenzhen Research Program (JCYJ20170818164704758, JCYJ20150925163005055), the Big Data Collaboration Research grant from SenseTime

Group (CUHK Agreement No. TS1610626), the Early Career Scheme of Hong Kong (No. 24204215), and ERC Advanced Grant VarCity. Limin Wang and Yuanjun Xiong equally contribute to this work.

REFERENCES

[1] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst.*, 2014, pp. 568–576.

[2] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 3551–3558.

[3] L. Wang, Y. Qiao, and X. Tang, "Motionlets: Mid-level 3D parts for human motion recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2013, pp. 2674–2681.

[4] 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. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 4694–4702.

[5] L. Wang, Y. Qiao, and X. Tang, "Action recognition with trajectory-pooled deep-convolutional descriptors," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 4305–4314.

[6] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. 25th Int. Conf. Neural Inf. Process. Syst.*, 2012, pp. 1106–1114.

[8] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Representations*, 2015, pp. 1–14.

[9] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 1–9.

[10] B. Zhou, À. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst.*, 2014, pp. 487–495.

[11] L. Shen, Z. Lin, and Q. Huang, "Relay backpropagation for effective learning of deep convolutional neural networks," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 467–482.

[12] 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.

[13] Y. Xiong, K. Zhu, D. Lin, and X. Tang, "Recognize complex events from static images by fusing deep channels," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 1600–1609.

[14] 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.*, vol. 126, no. 2/4, pp. 390–409, 2018.

[15] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 1725–1732.

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

[17] B. Zhang, L. Wang, Z. Wang, Y. Qiao, and H. Wang, "Real-time action recognition with enhanced motion vector CNNs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2718–2726.

[18] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.

[19] J. C. Niebles, C.-W. Chen, and F.-F. Li, "Modeling temporal structure of decomposable motion segments for activity classification," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 392–405.

[20] A. Gaidon, Z. Harchaoui, and C. Schmid, "Temporal localization of actions with actoms," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 11, pp. 2782–2795, Nov. 2013.

[21] L. Wang, Y. Qiao, and X. Tang, "Latent hierarchical model of temporal structure for complex activity classification," *IEEE Trans. Image Process.*, vol. 23, no. 2, pp. 810–822, Feb. 2014.

[22] B. Fernando, E. Gavves, M. José Oramas, A. Ghodrati, and T. Tuytelaars, "Modeling video evolution for action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 5378–5387.

[23] 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, 2016.

[24] G. Varol, I. Laptev, and C. Schmid, "Long-term temporal convolutions for action recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 6, pp. 1510–1517, 2018.

[25] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 2625–2634.

[26] H. Idrees, A. R. Zamir, Y.-G. Jiang, A. Gorban, I. Laptev, R. Sukthankar, and M. Shah, "The THUMOS challenge on action recognition for videos in the wild," *Comput. Vis. Image Understanding*, vol. 155, pp. 1–23, 2017.

[27] F. C. Heilbron, V. Escorcia, B. Ghanem, and J. C. Niebles, "Activitynet: A large-scale video benchmark for human activity understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 961–970.

[28] K. Soomro, A. R. Zamir, and M. Shah, "UCF101: A dataset of 101 human actions classes from videos in the wild," *CoRR*, vol. abs/1212.0402, 2012, http://arxiv.org/abs/1212.0402

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

[30] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, M. Suleyman, and A. Zisserman, "The kinetics human action video dataset," *CoRR*, vol. abs/1705.06950, 2017, http://arxiv.org/abs/1705.069502017.

[31] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[32] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2818–2826.

[33] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, "Temporal segment networks: Towards good practices for deep action recognition," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 20–36.

[34] D. A. Forsyth, O. Arikan, L. Ikemoto, J. F. O'Brien, and D. Ramanan, "Computational studies of human motion: Part 1, tracking and motion synthesis," *Found. Trends Comput. Graph. Vis.*, vol. 1, no. 2/3, pp. 77–254, 2005.

[35] P. K. Turaga, R. Chellappa, V. S. Subrahmanian, and O. Udrea, "Machine recognition of human activities: A survey," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 11, pp. 1473–1488, Nov. 2008.

[36] J. K. Aggarwal and M. S. Ryoo, "Human activity analysis: A review," *ACM Comput. Surv.*, vol. 43, no. 3, 2011, Art. no. 16.

[37] I. Laptev, "On space-time interest points," *Int. J. Comput. Vis.*, vol. 64, no. 2/3, pp. 107–123, 2005.

[38] G. Willems, T. Tuytelaars, and L. J. V. Gool, "An efficient dense and scale-invariant spatio-temporal interest point detector," in *Proc. Eur. Conf. Comput. Vis.*, 2008, pp. 650–663.

[39] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition via sparse spatio-temporal features," in *Proc. IEEE Int. Workshop Visual Surveillance Perform. Eval. Tracking Surveillance*, 2005, pp. 65–72.

[40] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, "Action recognition by dense trajectories," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2011, pp. 3169–3176.

[41] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, "Learning realistic human actions from movies," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2008, pp. 1–8.

[42] A. Kläser, M. Marszalek, and C. Schmid, "A spatio-temporal descriptor based on 3D-gradients," in *Proc. Brit. Mach. Vis. Conf.*, 2008, pp. 1–12.

[43] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *Proc. ECCV Workshop Statistical Learn. Comput. Vis.*, 2004, pp. 1–22.

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

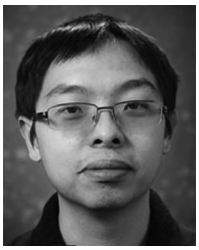
[45] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid, "Aggregating local image descriptors into compact codes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 9, pp. 1704–1716, Sep. 2012.
[46] Z. Cai, L. Wang, X. Peng, and Y. Qiao, "Multi-view super vector for action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 596–603.
[47] M. Raptis, I. Kokkinos, and S. Soatto, "Discovering discriminative action parts from mid-level video representations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 1242–1249.
[48] A. Jain, A. Gupta, M. Rodriguez, and L. S. Davis, "Representing videos using mid-level discriminative patches," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2013, pp. 2571–2578.
[49] W. Zhang, M. Zhu, and K. G. Derpanis, "From actemes to action: A strongly-supervised representation for detailed action understanding," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 2248–2255.
[50] J. Zhu, B. Wang, X. Yang, W. Zhang, and Z. Tu, "Action recognition with actons," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 3559–3566.
[51] S. Sadanand and J. J. Corso, "Action bank: A high-level representation of activity in video," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 1234–1241.
[52] L. D. Bourdev and J. Malik, "Poselets: Body part detectors trained using 3D human pose annotations," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2009, pp. 1365–1372.
[53] S. Ji, W. Xu, M. Yang, and K. Yu, "3D convolutional neural networks for human action recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 221–231, Jan. 2013.
[54] L. Sun, K. Jia, D. Yeung, and B. E. Shi, "Human action recognition using factorized spatio-temporal convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 4597–4605.
[55] Z. Wu, X. Wang, Y. Jiang, H. Ye, and X. Xue, "Modeling spatial-temporal clues in a hybrid deep learning framework for video classification," in *Proc. 23rd ACM Int. Conf. Multimedia*, 2015, pp. 461–470.
[56] J. Carreira and A. Zisserman, "Quo vadis, action recognition? A new model and the kinetics dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 4724–4733.
[57] Z. Qiu, T. Yao, and T. Mei, "Learning spatio-temporal representation with pseudo-3D residual networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 5534–5542.
[58] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 1933–1941.
[59] H. Pirsiavash and D. Ramanan, "Parsing videos of actions with segmental grammars," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 612–619.
[60] L. Wang, Y. Qiao, and X. Tang, "Video action detection with relational dynamic-poselets," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 565–580.
[61] P. F. Felzenszwalb, R. B. Girshick, D. A. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1627–1645, Sep. 2010.
[62] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. 32nd Int. Conf. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.
[63] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artif. Intell.*, vol. 17, no. 1/3, pp. 185–203, 1981.
[64] Y.-G. Jiang, J. Liu, A. Roshan Zamir, I. Laptev, M. Piccardi, M. Shah, and R. Sukthankar, "THUMOS challenge: Action recognition with a large number of classes," 2013, ICCV13-Action-Workshop/
[65] Y.-G. Jiang, J. Liu, A. Roshan Zamir, G. Toderici, I. Laptev, M. Shah, and R. Sukthankar, "THUMOS challenge: Action recognition with a large number of classes," 2014, http://crcv.ucf.edu/THUMOS14/
[66] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. B. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *CoRR*, vol. abs/1408.5093, 2014, http://arxiv.org/abs/1408.5093
[67] C. Zach, T. Pock, and H. Bischof, "A duality based approach for realtime TV-L^1 optical flow," in *Proc. 29th DAGM Symp. Pattern Recognit.*, 2007, pp. 214–223.
[68] A. Diba, A. M. Pazandeh, and L. Van Gool, "Efficient two-stream motion and appearance 3D CNNs for video classification," *CoRR*, vol. abs/1608.08851, 2016, http://arxiv.org/abs/1608.08851
[69] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," in *Proc. Brit. Mach. Vis. Conf.*, 2014, pp. 1–12.
[70] L. Wang, Y. Xiong, Z. Wang, and Y. Qiao, "Towards good practices for very deep two-stream convnets," *CoRR*, vol. abs/1507.02159, 2015, http://arxiv.org/abs/1507.02159
[71] C. Feichtenhofer, A. Pinz, and R. P. Wildes, "Spatiotemporal residual networks for video action recognition," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 3468–3476.
[72] H. Wang and C. Schmid, "LEAR-INRIA submission for the thumos workshop," in *Proc. ICCV Workshop THUMOS Challenge*, 2013, pp. 1–3.
[73] M. Jain, J. C. van Gemert, and C. G. Snoek, "What do 15,000 object categories tell us about classifying and localizing actions?" in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 46–55.
[74] Y. Zhu and S. Newsam, "Depth2Action: Exploring embedded depth for large-scale action recognition," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 668–684.
[75] 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 Understanding*, vol. 150, pp. 109–125, 2016.
[76] L. Wang, W. Li, W. Li, and L. Van Gool, "Appearance-and-relation networks for video classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 1430–1439.
[77] B. Ni, P. Moulin, X. Yang, and S. Yan, "Motion part regularization: Improving action recognition via trajectory group selection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 3698–3706.
[78] W. Zhu, J. Hu, G. Sun, X. Cao, and Y. Qiao, "A key volume mining deep framework for action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 1991–1999.
[79] Deep draw. (2016). [Online]. Available: https://github.com/auduno/deepdraw
[80] Z. Zhu, J. H. Engel, and A. Y. Hannun, "Learning multiscale features directly fromwaveforms," *Interspeech 2016, 17th Ann. Conf. Int. Speech Commun. Association*, San Francisco, CA, USA, Sep. 8–12, 2016, pp. 1305–1309.

Limin Wang received the BSc degree from Nanjing University, Nanjing, China, in 2011, and the PhD degree from the Chinese University of Hong Kong, Hong Kong, in 2015. From 2015 to 2018, he was a post-doctoral researcher with the Computer Vision Laboratory, ETH Zurich. He is currently a professor with the Department of Computer Science and Technology, Nanjing University. 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. He is a member of the IEEE.

Yuanjun Xiong received the BS degree from Tsinghua University, Beijing, China, in 2012, and the PhD degree in information engineering from the Chinese University of Hong Kong, Hong Kong, in 2016. His research interests include computer vision, machine learning, image understanding, and video content analysis. He worked as a postdoctoral fellow with the Multimedia Laboratory of the Chinese University of Hong Kong. He is currently a senior applied scientist at Amazon Web Services.

Zhe Wang received the BEng degree from the Beijing University of Posts and Telecommunications, Beijing, China, in 2014. He is currently working toward the PhD degree with the University of California, Irvine, Irvine. 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 received the PhD 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 more than 140 papers in journals and conference including, the *IEEE Transactions on Pattern Analysis and Machine Intelligence*, the *International Journal of Computer Vision*, the *IEEE Transactions on Image Processing*, 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. He is a senior member of the IEEE.



Dahua Lin received the BEng degree from the University of Science and Technology of China (USTC), in 2004, the MPhil degree from the Chinese University of Hong Kong (CUHK), in 2006, and the PhD degree from Massachusetts Institute of Technology (MIT), in 2012. He is an assistant professor with the Department of Information Engineering, the Chinese University of Hong Kong. Prior to joining CUHK, he served as a research assistant professor at Toyota Technological Institute at Chicago, from 2012 to 2014. He also worked in Microsoft Research for three times as an intern researcher, respectively in Beijing (2004), Redmond (2009), and Silicon Valley (2010). His research interest covers machine learning, computer vision, and big data analytics. In recent years, he primarily focused on structured deep learning, deep analysis of videos, as well as the use of deep learning techniques in probabilistic inference. He has published about fifty papers on top conferences and journals, e.g., ICCV, CVPR, ECCV, NIPS, and the *IEEE Transactions on Pattern Analysis and Machine Intelligence*. His seminal work on a new construction of Bayesian nonparametric models has won the best student paper award in NIPS 2010. He also received the outstanding reviewer award in ICCV 2009 and ICCV 2011.

Xiaoou Tang received the BS degree from the University of Science and Technology of China, Hefei, in 1990, the MS degree from the University of Rochester, Rochester, New York, in 1991, and the PhD degree from the Massachusetts Institute of Technology, Cambridge, in 1996. He is a professor and the chairman of the Department of Information Engineering, Chinese University of Hong Kong. He worked as the group manager of the Visual Computing Group, Microsoft Research Asia from 2005 to 2008. His research interests include computer vision, pattern recognition, and video processing. He received the Best Paper Award at the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2009 and Outstanding Student Paper Award at the AAAI 2015. He was a program chair of the IEEE International Conference on Computer Vision (ICCV) 2009 and served as an associate editor of the *IEEE Transactions on Pattern Analysis and Machine Intelligence* and the *International Journal of Computer Vision*. He is now the editor-in-chief of the *International Journal of Computer Vision*. He is a fellow of the IEEE.

Luc Van Gool received the degree in electromechanical engineering from the Katholieke Universiteit Leuven, in 1981. Currently, he is a professor with the Katholieke Universiteit Leuven in Belgium and the ETH in Zürich, Switzerland. He leads computer vision research at both places, where he also teaches computer vision. He has authored more than 200 papers in this field. He has been a program committee member of several major computer vision conferences. His main interests include 3D reconstruction and modeling, object recognition, tracking, and gesture analysis. He received several Best Paper awards. He is a co-founder of 5 spin-off companies. He is a member of the IEEE.