Learning temporal structures for human activity recognition

Tiantian Xu, Edward K Wong

Learning temporal structures for human activity recognition

Tiantian Xu
tiantian.xu@nyu.edu
Edward K. Wong
ewong@nyu.edu

Department of Computer Science and Engineering
Tandon School of Engineering
New York University
Brooklyn, New York, USA

Abstract

We propose a hierarchical method for learning temporal structures for the recognition of complex human activities or actions in videos. Low level features (HOG, HOF, MBH_x and MBH_y) are first computed from video snippets to form concatenated feature vectors. A novel segmentation algorithm based on K-means clustering is then used to divide the video into segments, with each segment corresponding to a sub-action with uniform motion characteristics. Using low level features as inputs, a many-to-one encoder is trained to extract generalized features for the snippets in each segment. A second many-to-one encoder is then used to compute higher-level features from the generalized features. The higher-level features from individual segments are then concatenated together and used to train a third many-to-one encoder to extract a high-level feature representation for the entire video. The final descriptor is the concatenation of higher-level features from individual segments and the high-level feature for the entire video. Using the proposed descriptor and a mutli-class linear support vector machine (SVM), we achieved state-of-the-art results on datasets Olympic Sports and UCF50, and beat the state-of-the-art result on the challenging HMD51 dataset by a wide margin of 17%.

1 Introduction

Human action recognition in videos is one of the most studied topics in computer vision[1, 19] with applications in video surveillance, video annotation and retrieval, sports action recognition, etc. Despite recent advances, action recognition remains a challenging problem. This is especially true when it comes to recognizing complex actions that consist of multiple sub-actions. For example, the action *long jump* as shown in Fig 1 consists of the sub-actions *run*, *jump* and *slide*. In fact, sub-action *run* can be further decomposed into smaller sub-actions *speed up*, *run at even pace* and *slow down*.These sub-actions and their temporal ordering together define the action.

In this paper, we propose a hierarchical method for learning temporal structures for the recognition of complex human activities or actions in videos. A video is automatically divided into segments, with each segment corresponding to a sub-action with uniform motion characteristics. In each segment, we compute low level features from video snippets that are made up of a short sequence of consecutive frames. Higher level features are then generated from the low level features using two layers of many-to-one encoders. A third many-to-one

Figure 1: Frames from action *long jump*. Clearly it goes through three distinct sub-actions: *run*, *jump* and *slide*.

encoder is then used to generate a global representation from the higher level features of individual segments.

Automatically decomposing a complex action in a video into segments is a difficult task due to the inherent ambiguity in defining sub-actions and the segment boundaries between them. Using fixed-sized segments is not a viable approach since actions and sub-actions have variable lengths (number of frames) in different videos. We propose an unsupervised segmentation method to decompose videos into segments, based on K-means clustering of low-level features extracted from video snippets.

In Section 2, we give a review of related work. In Section 3, we give background information on the many-to-one encoder architecture. Section 4 gives details of our unsupervised temporal segmentation algorithm. Section 5 describes our method for high-level feature extraction using many-to-one encoders. In Section 6, we describe our experimental results, and in Section 7, we give our conclusion. The contributions of our paper are:

- We have developed an unsupervised temporal segmentation method to decompose complex actions in videos into segments.
- We have developed a hierarchical method to learn higher level features from individual segments and the video as a whole, and then combine them to form a global representation.
- Our video representation is compact and discriminative, achieving state-of-the-art recognition accuracy on benchmark datasets Olympic Sports and UCF50, and beating the state-of-the-art result [11] on the challenging dataset HMDB51 by over 17%.

2 Related Work

Video representation is arguably the most important part of any action recognition method. Hand-crafted representations such as Bag-of-Words or Fisher vector combined with spatial-temporal interest points or dense trajectories have demonstrated strong performance in action recognition [8, 27, 28]. However, they ignore the decomposition of complex actions into sub-actions and therefore do not perform well in real world videos with complex actions. For example, recognition accuracy on one of the most challenging datasets HMDB51, dominated by real and complex actions, using iDT+fisher vector is less than 60% [27].

Encouraged by the successes in the use of deep learning techniques (e.g. Convolution Neural Networks (CNN)) for object detection and classification in 2D images, researchers have tried to extend deep learning to the video domain[3, 21, 32]. However, a major challenge for applying deep learning methods to videos is their big GPU memory requirements.

Although some work [11, 13, 31] have addressed this problem, the marginal advantages gained by using deep learning methods are offset by the computational complexity.

Comparing to CNNs, many-to-one encoders are less training-intensive. First proposed in [35], a many-to-one encoder learns high-level features in a supervised manner by reducing intra-class variances. More specifically, the encoder is a feed-forward network where input instances of large intra-class variations are mapped to the same target output. Encouraged by its successes in producing compact and discriminative features in pose-invariant face recognition [35], 3D object retrieval [2] and cross-dataset action recognition [34], in this work, we adopt the many-to-one encoder to cope with the high intra-class variations in human action recognition. In our proposed method, a set of many-to-one encoders are arranged hierarchically to extract generalized and high level features at different temporal scales: snippets, segments, and then the video as a whole.

Hierarchical feature representation has been used to model motion dynamics at different levels of temporal granularity. Some works build hierarchy with layers of latent states [10, 18, 22, 24, 30]. [30] proposed to model complex actions using a latent hierarchical tree-like model. Bag-of-words representation is used to represent video segments and the entire video in a bottom-up manner. In other approaches, hierarchy is captured explicitly by using spatial-temporal context or video segmentation [4, 17, 23, 25, 26, 29]. The pioneer work of [33] approached temporal segmentation by detecting 'rest states', which are temporal locations where one action finishes and another begins, under the assumption that the rest states correspond to little or no motion. Although this approach may work for multi-action sequences generated in a controlled setting, it is not clear how it could adapt to complex actions in the real world when the assumption for rest states (little or no motion) between sub-actions no long holds. More recently, [31] incorporated temporal segments with CNN to address the limited usage of temporal information in deep neural network based approaches. They evenly partitions a video into fixed-length segments whereas, in our method, we perform temporal segmentation and generate segments of variable lengths and each with similar motion characteristics. Another key difference between our method and [31] is that the latter uses RGB and optical flow features whereas we use HOG, HOF, MBH_x and MBH_y, which are more discriminative for action recognition. [23] proposed to decompose and encode videos in an unsupervised manner. However, low-level features are extracted at the frame level and, as a result, the video representation contains no temporal information.

3 Many-to-One Encoders

In this section, we present background on the many-to-one encoder and its training procedure. We will describe details on the encoder inputs in Sections 4.1 and 5.

Many-to-one encoder is a feed-forward neural network that is trained to map multiple inputs to the same target output. A typical many-to-one encoder has three layers: an input layer, a hidden layer and an output layer. As demonstrated in [2, 34, 35], when forced to map instances from the same class to the class signature, it can extract high level and discriminative features in the hidden layer. The encoder learns to reduce the variances between instances of the same class while maintaining the differences between different classes. In this work, we use the class centroid (in truncated form) as the class signature.

Let x_i^c denotes the i-th instance of class c and y^c the class signature of class c. Given training samples $\{(x_i^c, y^c)\}$, where $|x_i^c| = L$ and $|y^c| = O$, a many-to-one encoder is defined by the mappings $f_1 \colon \mathbb{R}^L \to \mathbb{R}^H$ and $f_2 \colon \mathbb{R}^H \to \mathbb{R}^O$, where H is the dimension of the hidden

Figure 2: Illustration of temporal segmentation process. Numbers are cluster labels assigned to each snippet after K-means clustering. Top: evenly divide the video into K segments. Bottom: segmentation result after updating segment boundaries (red lines) such that each segment is dominated by one cluster label. Afterwards, segments are denoted as segment 1, segment 2,..., segment K (from left to right).

layer. The network parameters are learned by minimizing the following loss function:

$$E = \frac{1}{2N}\sum_{i,c}\|y^c - h(x_i^c, W, b)\|_2^2 + \frac{\lambda}{2}\|W\|^2 \tag{1}$$

where $h(\cdot) = f_2(f_1(\cdot))$, W and b are the weights and biases of the network, λ is the parameter of the regularization term, and N is the number of training instances. The regularization term prevents over-fitting by penalizing large $l2$ norm of the weights of the network. Training is done via stochastic gradient descent, where the objective function is minimized by iteratively updating the weights and biases [9].

After training, the encoder part (input layer to hidden layer) can be used to generate generalized features of the input samples.

4 Unsupervised Temporal Segmentation

We propose an unsupervised temporal segmentation method based on K-means clustering. The objective is to produce K non-overlapping temporal segments so that motion within each segment is homogeneous.

4.1 Low-level features

Instead of using frames as the basic units for segmentation, as in many previous works, we use snippets of consecutive frames as basic units. The rationale being that low level features computed from snippets of frames contain richer temporal information. Given a video, a set of overlapping snippets are produced by sliding a fixed-sized temporal window, using a certain step size, over the video. Low-level features are then computed from each snippet. In this work, we use the widely-used features HOG, HOF, MBH_x and MBH_y [27]. These features are extracted in dense non-overlapping spatial-temporal blocks that cover the entire snippet. The four features, computed from individual blocks, are then concatenated to form a single feature vector for representing the snippet.

4.2 Clustering and forming segments

Our unsupervised segmentation procedure is summarized in Algorithm 1 below. We first perform K-means clustering on the set of snippets in the feature space and then assign each

snippet a label based on the cluster it belongs to. The video is initially divided into K equal-size segments with $K-1$ segment boundaries. The segment boundaries are iteratively adjusted such that each segment is dominated by one of the cluster labels through majority vote, while maintaining the constraint that there is at least one snippet in each segment. This process is illustrated in Fig 2.

Algorithm 1: Unsupervised temporal segmentation.

Input : Feature vectors for snippets $S = \{s_i\}$ and number of segments K
; `// Snippets are indexed by starting frame number`
Output: Segment boundaries $P = \{p_k\}$, where p_k is the index of the last snippet in the j-th segment.

Divide video into K even partitions and initialize P
Perform K-means clustering on snippet features. Assign cluster label l_i to s_i.
Initialize candidate set $C = \{1, 2, ..., K\}$
for $k \in 1, 2, ..., K-1$ **do**
    Compute dominant cluster D_k in current partition
    while $p_k > p_{k-1}$ *and* $D_k \notin C$ **do**
        Decrease p_k by one
        Update D_k by performing majority vote on snippet labels in current partition
    end
    Remove D_k from C
    if $l_{p_k} \neq D_k$ **then**
        Decrease p_k until $l_{p_k} = D_k$
    else
        Increase p_k until $l_{p_k} \neq D_k$ or $p_k = p_{k+1} - 1$
    end
end

5 High-level Features Extraction Using Many-to-one Encoders

In this section, we describe our method of using many-to-one encoders to extract high-level features from low-level features computed from video snippets. High-level features are generated for individual segments (corresponding to sub-actions) as well as for the video as a whole (corresponding to the action as a whole.)

Since we are dealing with human action recognition, we spatially divide the video into the upper and lower partitions, loosely corresponding to the upper and lower body of the actor. This allows us to capture the motion of the upper and lower body separately for a more refined representation. Our method does not divide the video into smaller spatial partitions as in the spatial-pyramid-based approach [8] because smaller partitions are not necessarily semantically meaningful. For example, a 2×2 spatial partition is meaningful when the person is facing the camera so that each partition contains one of the limbs, but the partitioning may not be meaningful when the camera is looking at the person on the side.

As a result of dividing the video into two partitions, the snippets will also be spatially divided into two sub-snippets (upper half and lower half). We compute the four types of

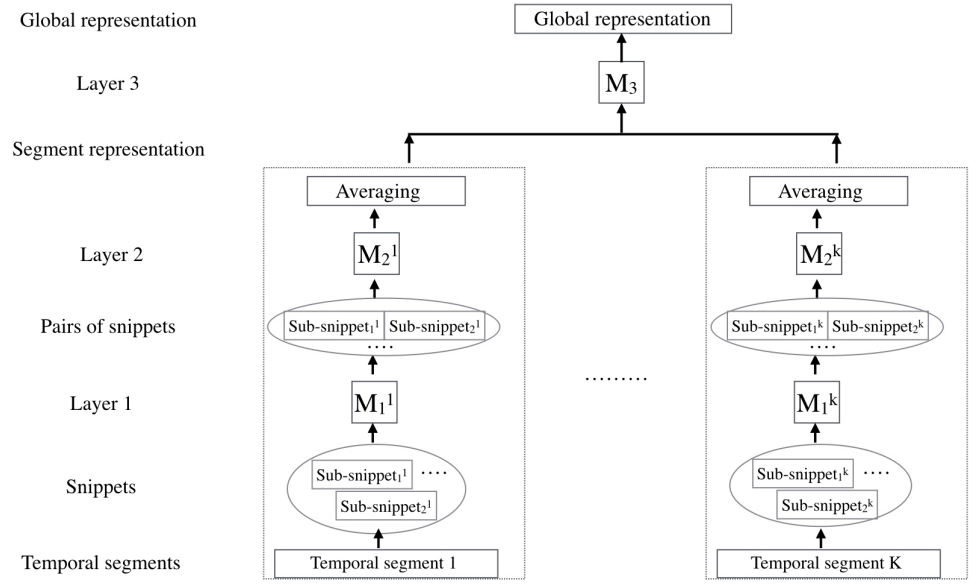


Figure 3: Proposed hierarchical architecture. Each *M* represents a many-to-one encoder. For visual clarity, only one set of encoders among eight (2 spatial partitions x 4 low-level feature types) are shown.

low-level features (HOG, HOF, MBH_x and MBH_y) for each sub-snippet. We then train a set of many-to-one encoders to generate high-level features for the individual segments and a high-level feature to represent the video as a whole.

In each video segment, we use the low-level features computed from the sub-snippets to train a set of many-to-one encoders. For each feature type, a many-to-one encoder is trained for each of the two spatial partitions. After training, the encoder part of the network is used to generate generalized features from the low-level features of individual sub-snippets. This is represented as layer 1 in Figure 3. The generalized features are of size H_1 where H_1 is the size of the hidden layer of the many-to-one encoder in layer 1.

The generalized features from layer 1 are then used to train a second many-to-one encoder in layer 2. We concatenate the generalized features of two consecutive sub-snippets to form the input to the many-to-one encoder in layer 2. The network in layer 2 therefore encodes over a sequence of frames that is twice as long as that in layer 1. After training, the encoder part of the network is used to generate higher-level features from the generalized features from layer 1. We compute the average of the higher-level features of the sub-snippets and use the average to represent the video segment. The higher level representation of each segment consists of two feature vectors, one for each spatial partition. The dimensions of the feature vectors are H_2, where H_2 is the size of the hidden layer of the encoder in layer 2.

High-level features from the K temporal segments are concatenated to form the input to a third many-to-one encoder in layer 3. Segment features computed from a set of training videos are used to train the third many-to-one encoder. After training, the encoder part of the network can be used to generate a high-level feature for the video as a whole.

For each of the four low-level feature types, we concatenate the high-level features from individual segments and the high-level feature for the video as a whole. Finally, we con-

catenate the high-level features from the four feature types to form the final video descriptor. The final descriptor is a vector of length $4 \cdot 2 \cdot (KH_2 + H_3)$ where K is the number of temporal segments, H_2 and H_3 are the size of the hidden layers of the encoders in layers 2 and 3, respectively. Using the video descriptors generated from the training videos, we train a multi-class linear SVM for classification.

6 Experiments and Results

6.1 Datasets

The **Olympic Sports** dataset [14] contains 16 actions with a total of 783 videos. The videos were taken from YouTube and each video shows an athlete performing a sport. This is a challenging dataset where videos are subject to varying viewpoints, scale, background and lighting conditions. We followed [14] and evaluate on the videos using a train/test split of 649 and 134 videos and report the mean average precision.

The **UCF50** dataset [20] features 50 real-life activities, including sports as well as daily life exercises. They contain consumer videos taken from YouTube and have significant intra-class variance in the background, viewpoints, and lighting conditions. The videos are divided into 25 groups. Each group contains 50 actions with at least 4 videos per action. There is a total of 6,618 videos. We use the leave-one-group-out evaluation scheme as suggested by the authors and report the average class accuracy.

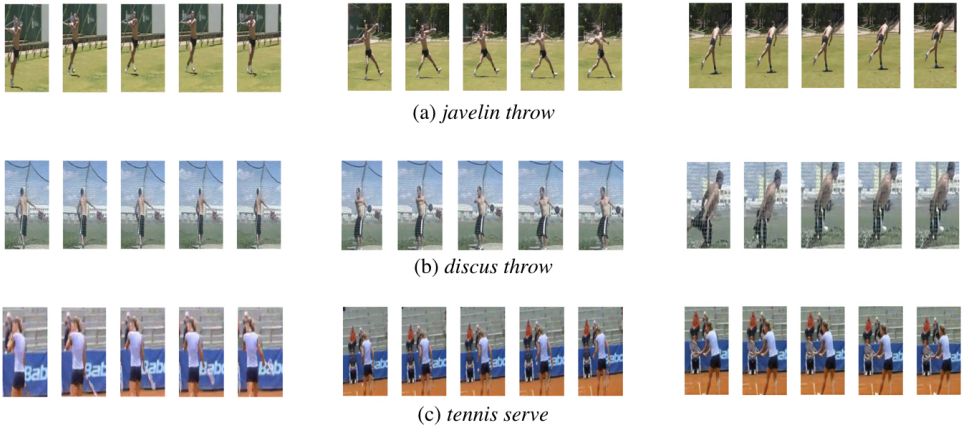
The **HMDB51** dataset [8] is a challenging large-scale action dataset with 51 classes and 6,766 video clips. Actions in this dataset is very diverse, ranging from daily activities (*brush hair*) to sports (*golf*) and human interactions (*hug*). This dataset consists of movie clips as well as consumer video clips. We follow the three-way train/test split evaluation scheme recommended by the authors, with each action having about 70 training videos and 30 testing videos. We report the average class accuracy in our results. For all datasets, we use the bounding box information provided by authors of [27] at the provided url[1].

6.2 Experimental set-up

In our experiments, we use a snippet length of 7 frames and a step size of 1 frame. Frames are cropped and reduced to a size of 128×64 pixels using the bounding box information provided by the authors of [27]. We adopt the same low-level feature extraction parameters as in [27]. The raw snippet features are reduced to 100 via PCA dimensionality reduction. We tried different hidden layer sizes and the following setting consistently produces the best recognition accuracy: $H_1 = 100$, $H_2 = 50$ and $H_3 = 50$. The input dimensions for the three networks are 100, 200 and $50 * K$, respectively. The output dimensions are set to 50. In training, the learning rate is 0.001, the momentum is 0.9 and λ is 0.0001. Training is done over 50 iterations. We initialized the weights and biases in the networks with random numbers that range from 0 to 1. We used the *tanh* function as the activation function. We experimented with different values of K for temporal segmentation and report the results in Section 6.3. Our implementation of the many-to-one encoder training procedure can be found at the provided url [2].

[1] http://lear.inrialpes.fr/~wang/improved_trajectories

[2] https://www.mathworks.com/matlabcentral/fileexchange/63685-many-to-one-encoder-training

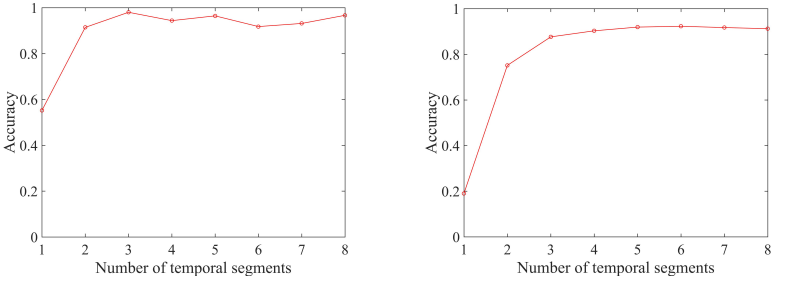


(a) *javelin throw*

(b) *discus throw*

(c) *tennis serve*

Figure 4: Illustration of temporal segmentation results.



(a) Olympic Sports (b) HMDB51

Figure 5: Recognition accuracy vs. number of temporal segments for the Olympic Sports and HMDB51 datasets. Accuracy measure used for Olympic Sports dataset is mean average precision and measure used for HMDB51 is average class accuracy.

6.3 Results and analysis

Visual appearance of the segments We present temporal segmentation results on three videos from the Olympic Sports dataset in Fig 4. For clarity, only the first five frames in each segment are shown. It can be seen that our method can form segments of distinct visual appearances: *javelin throw* is decomposed into *lift arm to throw*, *throw* and *done throw*; *discus throw* is decomposed into *rotate body*, *prepare to throw* and *rotate to throw* and *tennis serve* is decomposed into *walk*, *take position* and *serve*. As will be shown below, our temporal segmentation algorithm contributes to the superior recognition results of our method.

Number of segments Fig 5 illustrates the change in recognition accuracy versus number of temporal segments for the Olympic Sports and HMDB51 datasets. For both datasets, performing temporal segmentation and encoding video at multiple temporal granularity drastically improves recognition accuracy. However, increasing the number of segments beyond a certain point does not necessarily improves performance, likely due to the underlying dynamics of the actions. We observe that accuracy flattens out at $K = 3$ for both datasets. This is inline with the findings in works such as [30, 31]. We also experimented with using fixed

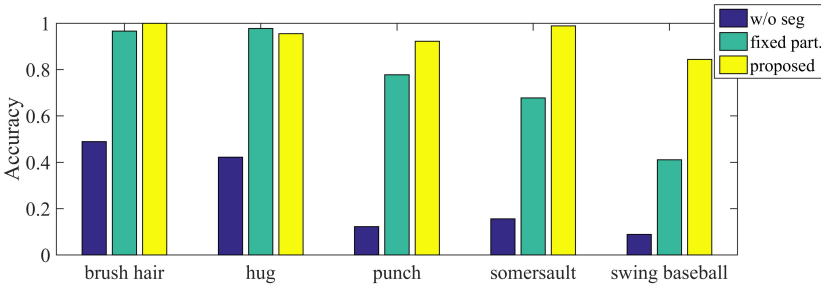


Figure 6: Accuracy for selected actions from the HMDB51 dataset with no temporal segmentation (blue), fixed partitioning (green) and proposed segmentation (yellow). Accuracy measure used is average class accuracy.

partitioning as in [31]. While the accuracy is comparable to using the proposed temporal segmentation on the Olympic Sports dataset, a 7% decrease in accuracy is observed on the HMDB51 dataset. This suggests that complex real world datasets such as the HMDB51 would benefit from temporal segmentation. Comparing the effects of temporal segmentation on the Olympic Sports dataset and the HMDB51 dataset, we observe a much larger performance boost on the HMDB51 dataset from K=1 to K=2. We conjecture that this is because videos in Olympic Sports dataset are less complex than those of HMDB51, which makes them easier to classify even without temporal segmentation.

Effect of temporal segmentation Fig 6 shows the recognition accuracy on five actions from the HMDB51 dataset using video descriptors constructed from (1) no temporal segmentation, where the video descriptor is the average snippet features from layer 2, (2) fixed partitioning, where each segment has the same length, and (3) the proposed segmentation scheme. On all actions, both fixed partitioning and the proposed segmentation scheme result in higher recognition accuracy than no temporal segmentation. For *brush hair* and *hug*, fixed partitioning and the proposed segmentation scheme have about the same accuracy. This is probably due to the fact that these two actions have little temporal structures, which also explains the relatively smaller improvement over no segmentation. On the other hand, for actions with distinct stages, such as *punch*, *somersault* and *swing baseball*, the proposed segmentation scheme has much better accuracy than the other two methods. It is also worth noting that the fixed partitioning scheme has reasonably good performance, which proves the effectiveness of our proposed hierarchical encoder architecture and also confirms the findings in [31].

Evaluate various architectural settings We evaluated the performance change on the Olympic Sports dataset using the following architectural settings: without snippet concatenation in the input to layer 2, using layer 2 features alone and layer 3 features alone, and using the four types of low-level features separately. With K = 3, using layer 2 and layer 3 features alone reduce the recognition accuracy by about 2% and 5%, respectively. Removing snippet concatenation in the input to layer 2 reduces the accuracy by about 7%. This demonstrates the effectiveness of the proposed hierarchical architecture for extracting features at multiple temporal granularity. Table 1 shows the recognition accuracy using the four types of low-level features separately as well as combined. Evidently, as the four types of features capture complementary appearance (HOG) and motion (HOF, MBH_x, and MBH_y) characteristics, combining the four results in a more discriminative representation than using

	HOG	HOF	MBH_x	MBH_y	Combined
Olympic Sports (mAP%)	76.7	83.1	89.3	80.9	99.1

Table 1: Recognition accuracy using high level features learned from the four feature types separately and combined on the Olympic Sports dataset with K = 3 (mAP = mean average precision.)

Olympic Sports (mAP%)		UCF50 (mAcc%)		HMDB51 (mAcc%)	
Wang and Schmid [27]	91.1	Wang and Schmid [27]	91.2	Wang and Schmid [27]	57.2
Lan *et al.* [9]	91.4	Oneata *et al.* [15]	90.0	Lan *et al.* [9]	65.1
Peng *et al.* [16]	93.8	Narayan and Ramakrishnan [13]	89.4	Wang *et al.* [31]	68.5
Li *et al.* [10]	96.6	Lan *et al.* [9]	94.4	Lan and Hauptmann [8]	75.0
Ours with $K = 3$	**99.1**	Ours with $K = 6$	**96.8**	Ours with $K = 6$	**92.3**

Table 2: Comparison of our results to state-of-the-art methods. The best accuracies are highlighted in bold. Our results beat the state-of-the-art methods on all three datasets, most notably by over 17% on HMDB51, one of the most challenging benchmark datasets (mAP = mean average precision and mAcc = average class accuracy.)

the features separately. This is consistent with the finding in previous work such as [27].

6.4 Comparison to state-of-the-art

Table 2 compares our results against the state-of-the-art on all three datasets. Our method beats the state of the art on the Olympic Sports and UCF50 datasets by 2-3%. This is already quite significant since the state-of-the-art results on both datasets are already quite high (above 94%). The most significant gain is on HMDB51, where our result beats the state-of-the-art by over 17%. Notably, most of the state-of-the-art methods on HMDB51 are CNN-based. Our method is much less training intensive because different spatial partitions, feature types, and temporal segments can be trained in a parallel manner. For example, it only takes around 16 minutes to finish training on the HMDB51 dataset, comparing to 9 hours training time as reported in [31]. Another advantage of our method over CNN-based methods is the low dimensionality of the video descriptor: 1,600 versus thousands or tens-of-thousands in deep neural networks.

7 Conclusion

We have presented a novel temporal segmentation method and hierarchical architecture, consisting of an ensemble of many-to-one encoders, for learning generalized features from action videos at multiple temporal granularity. Our video descriptor is highly compact and discriminative and beat the state-of-the art results on challenging datasets. While the proposed method achieves superior performance comparing with CNN-based methods, it is possible to extract more powerful features by combining the two; for example, train many-to-one encoders on features extracted by CNNs.

References

[1] Jake K Aggarwal and Michael S Ryoo. Human activity analysis. *ACM Comput. Surv.*, 43(3):1–43, 2011. ISSN 03600300. doi: 10.1145/1922649.1922653.

[2] Yi Fang, Jin Xie, Guoxian Dai, Meng Wang, Fan Zhu, Tiantian Xu, and Edward Wong. 3D Deep Shape Descriptor. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015.

[3] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Fei Fei Li. Large-scale video classification with convolutional neural networks. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014. ISSN 10636919. doi: 10.1109/CVPR.2014.223.

[4] Adriana Kovashka and Kristen Grauman. Learning a hierarchy of discriminative space-time neighborhood features for human action recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2046–2053, 2010. ISSN 10636919. doi: 10.1109/CVPR.2010.5539881.

[5] Hildegard Kuehne, Hueihan Jhuang, Est{\'\i}baliz Garrote, Tomaso Poggio, and Thomas Serre. HMDB: A large video database for human motion recognition. *Proceedings of the IEEE International Conference on Computer Vision*, pages 2556–2563, 2011. ISSN 1550-5499. doi: 10.1109/ICCV.2011.6126543.

[6] Zhenzhong Lan and Alexander G Hauptmann. Deep Local Video Feature for Action Recognition. *arXiv preprint*, 2017.

[7] Zhenzhong Lan, Ming Lin, Xuanchong Li, Alexander G Hauptmann, and Bhiksha Raj. Gaussian Pyramid: Multi-skip Feature Stacking for Action Recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015.

[8] Ivan Laptev, Marcin Marszalek, Cordelia Schmid, and Benjamin Rozenfeld. Learning realistic human actions from movies. In *IEEE Conference on Computer Vision and Pattern Recognition,*, 2008. ISBN 9781424422432. doi: 10.1109/CVPR.2008.4587756.

[9] Yann LeCun, Leon Bottou, Genevieve B. Orr, and Klaus-Robert Muller. Efficient BackProp. *Neural networks: Tricks of the trade. Springer Berlin Heidelberg*, 75:9–48, 1988.

[10] Kang Li, Jie Hu, and Yun Fu. Modeling complex temporal composition of actionlets for activity prediction. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7572 LNCS (PART 1):286–299, 2012. ISSN 03029743. doi: 10.1007/978-3-642-33718-5_21.

[11] Yingwei Li, Weixin Li, Vijay Mahadevan, and Nuno Vasconcelos. VLAD 3 : Encoding Dynamics of Deep Feature for Video Recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016.

[12] Sanath Narayan and Kalpathi R Ramakrishnan. A Cause and Effect Analysis of Motion Trajectories for Modeling Actions. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2014.

[13] Joe Yue-hei Ng, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George TOderici. Beyond Short Snippets : Deep Networks for Video Classification. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015. ISSN 10636919. doi: 10.1109/CVPR.2015.7299101.

[14] Juan Carlos Niebles, Chih Wei Chen, and Li Fei-Fei. Modeling temporal structure of decomposable motion segments for activity classification. *European Conference on Computer Vision*, 6312 LNCS(PART 2):392–405, 2010. ISSN 03029743. doi: 10.1007/978-3-642-15552-9_29.

[15] Dan Oneata, Jakob Verbeek, and Cordelia Schmid. Action and Event Recognition with Fisher Vectors on a Compact Feature Set. *IEEE International Conference on Computer Vision*, pages 1817–1824, 2013. ISSN 1550-5499. doi: 10.1109/ICCV.2013.228.

[16] Xiaojiang Peng, Changqing Zou, Yu Qiao, and Qiang Peng. Action recognition with stacked fisher vectors. *European Conference on Computer Vision*, 8693 LNCS(PART 5):581–595, 2014. ISSN 16113349. doi: 10.1007/978-3-319-10602-1_38.

[17] Juan-Manuel Perez-Rua, Tomas Crivelli, Patrick Perez, and Patrick Bouthemy. Discovering motion hierarchies via tree-structured coding of trajectories. *British Machine Vision Conference*, pages 1–12, 2016.

[18] Víctor Ponce-López, Hugo Jair Escalante, Sergio Escalera, and Xavier Baró. Gesture and Action Recognition by Evolved Dynamic Subgestures. *British Machine Vision Conference*, pages 1–13, 2015.

[19] Ronald Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 28(6):976–990, 2010. ISSN 02628856. doi: 10.1016/j.imavis.2009.11.014.

[20] Kishore K. Reddy and Mubarak Shah. Recognizing 50 human action categories of web videos. *Machine Vision and Applications*, 24(5):971–981, 2013. ISSN 09328092. doi: 10.1007/s00138-012-0450-4.

[21] Karen Simonyan and Andrew Zisserman. Two-Stream Convolutional Networks for Action Recognition in Videos. *arXiv preprint arXiv:1406.2199*, pages 1–11, 2014. ISSN 1098-6596. doi: 10.1017/CBO9781107415324.004.

[22] Yale Song, Louis Philippe Morency, and Randall Davis. Action recognition by hierarchical sequence summarization. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3562–3569, 2013. ISSN 10636919. doi: 10.1109/CVPR.2013.457.

[23] Bing Su, Jiahuan Zhou, Xiaoqing Ding, Hao Wang, and Ying Wu. Hierarchical Dynamic Parsing and Encoding for Action Recognition. *European Conference on Computer Vision*, 9905:202–217, 2016. ISSN 0302-9743. doi: 10.1007/978-3-319-46448-0.

[24] Kevin Tang, Li Fei-Fei, and Daphne Koller. Learning latent temporal structure for complex event detection. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1250–1257, 2012. ISSN 978-1-4673-1228-8. doi: 10.1109/CVPR.2012.6247808.

[25] Ekaterina H Taralova, Fernando De La Torre, and Martial Hebert. Motion Words for Videos. *European Conference on Computer Vision*, pages 725–740, 2014.

[26] Vinay Venkataraman, Ioannis Vlachos, and Pavan Turaga. Dynamical Regularity for Action Analysis. *British Machine Vision Conference*, 2015.

[27] Heng Wang and Cordelia Schmid. Action Recognition with Improved Trajectories. *IEEE International Conference on Computer Vision*, pages 3551–3558, 2013. ISSN 1550-5499. doi: 10.1109/ICCV.2013.441.

[28] Heng Wang, Muhammad Muneeb Ullah, A Kläser, Ivan Laptev, Cordelia Schmid, Alexander Klaser, Ivan Laptev, and Cordelia Schmid. Evaluation of local spatio-temporal features for action recognition. *Procedings of the British Machine Vision Conference 2009*, pages 124.1–124.11, 2009. doi: 10.5244/C.23.124.

[29] Jiang Wang, Zhuoyuan Chen, and Ying Wu. Action recognition with multiscale spatio-temporal contexts. *EEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3185–3192, 2011. ISSN 10636919. doi: 10.1109/CVPR.2011.5995493.

[30] Limin Wang, Yu Qiao, and Xiaoou Tang. Latent Hierarchical Model of Temporal Structure for Complex Activity Classification. *Image Processing, IEEE Transactions on*, 23(2):810–822, 2014. ISSN 1057-7149. doi: 10.1109/TIP.2013.2295753.

[31] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal Segment Networks: Towards Good Practices for Deep Action Recognition. *European Conference on Computer Vision*, 2016. doi: 10.1007/978-3-319-46484-8_2.

[32] Yifan Wang, Jie Song, Limin Wang, Luc Van Gool, and Otmar Hilliges. Two-Stream SR-CNNs for Action Recognition in Videos. *British Machine Vision Conference*, 2016.

[33] Daniel Weinland, Remi Ronfard, and Edmond Boyer. Automatic discovery of action taxonomies from multiple views. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2:1639–1645, 2006. ISSN 10636919. doi: 10.1109/CVPR.2006.65.

[34] Tiantian Xu, Fan Zhu, Edward K. Wong, and Yi Fang. Dual many-to-one-encoder-based transfer learning for cross-dataset human action recognition. *Image and Vision Computing*, 55:127–137, 2016. ISSN 02628856. doi: 10.1016/j.imavis.2016.01.001.

[35] Yizhe Zhang, Ming Shao, Edward K. Wong, and Yun Fu. Random faces guided sparse many-to-one encoder for pose-invariant face recognition. *IEEE International Conference on Computer Vision*, pages 2416–2423, 2013. ISSN 1550-5499. doi: 10.1109/ICCV.2013.300.