LETTER

Self-Supervised Learning of Video Representation for Anticipating Actions in Early Stage

Yinan LIU[†a)], ***Member***, **Qingbo WU**[†], ***Nonmember***, **Liangzhi TANG**[†], ***and*** **Linfeng XU**[†b)], ***Members***

SUMMARY In this paper, we propose a novel self-supervised learning of video representation which is capable to anticipate the video category by only reading its short clip. The key idea is that we employ the Siamese convolutional network to model the self-supervised feature learning as two different image matching problems. By using frame encoding, the proposed video representation could be extracted from different temporal scales. We refine the training process via a motion-based temporal segmentation strategy. The learned representations for videos can be not only applied to action anticipation, but also to action recognition. We verify the effectiveness of the proposed approach on both action anticipation and action recognition using two datasets namely UCF101 and HMDB51. The experiments show that we can achieve comparable results with the state-of-the-art self-supervised learning methods on both tasks.

key words: *action anticipation, video frame encoding, convolutional neural network*

1. Introduction

Action anticipation [1], aims to find out the action categories of a given video when only its small fraction is available, which is common in various streaming video services. In the real-world applications, action anticipation can be used in the situation such as autonomous navigation, sports analysis and forecasting dangerous. However, teaching the machine to anticipate video is not an easy task. The traditional supervised approaches tend to manually annotate videos for each segment, or crop the video frames to ensure consistency, which are often very time consuming. To develop a cost-efficient representation for the video, recent researches [2], [3] tend to explore the spatial-temporal information from unlabeled video. Following this objective, in this paper, we propose a video representation learning method based on the self-supervised learning paradigm [4]. The self-supervised learning treats the structure or relationship of the videos as a supervisory signal (*e.g.*, video temporal order, appearance similarity), thus it can use the supervised manner to train without video labels. Our main goal in this work is to anticipate the action categories of a video in early stage. Thus we need to model the relationship between videos and sub-videos. The core idea behind our work is: by employing different video frame encoding techniques, we could compress arbitrary length videos into fixed-sized images, these images will then be embedded into feature space by the Siamese networks [4], [5]. In the feature space, the sub-videos from the same video should be close (similar) to its neighbor, and the video should be close to its sub-video as well, otherwise, they should be far from each other. The network is designed based on the above assumptions. Then we design a motion-based temporal segmentation to refine the training process. In the end, our approach can learn representative features which indicate the relationship of videos and sub-videos in a self-supervised manner. The learned video representation can then be used in both action anticipation and action recognition. In the remainder of this paper, we first illustrate the core idea of the proposed approach, then we given details on how we process the video frames. Next we present the self-supervised learning framework, and show how we can improve the training by designing a motion-based temporal segmentation strategy. The proposal is evaluated by two challenging datasets UCF101 [6] and HMDB51 [7].

Manuscript received January 18, 2018.
Manuscript publicized February 21, 2018.

[†]The authors are with the School of Electronic Engineering, University of Electronic Science and Technology of China, Chengdu, China.
a) E-mail: yates2012codec@126.com
b) E-mail: lfxu@uestc.edu.cn

DOI: 10.1587/transinf.2018EDL8013

2. Self-Supervised Learning of Video Representation

The main idea behind our self-supervised learning is shown in Fig. 1. We model the learning process as two different self-supervised learning tasks. Both of these tasks take encoded video pairs as inputs. The work contains three components: first we convert the arbitrary length videos to a fixed-sized data structure for the network, we employ three different video frame encoding techniques to achieve this. Second we build a network to accomplish our tasks by learning from the videos. We employ a Siamese architecture, which perfectly fits the tasks in our work. Third, by utilizing the motion information from the videos, we give the idea on how to refine the training process.

2.1 Video Frame Encoding

In our work, we employ three different video frame encoding strategies. We adopt them to the videos and sub-videos.

2.1.1 Sum of Frame Difference Encoding

Given a video $\mathbf{V} = \{I_1, \ldots, I_T\}$, where I_t is the t-th frame of the $\mathbf{V}$. We first compute the difference of consecutive frames $\mathbf{d}_t = I_{t+1} - I_t$ in video $\mathbf{V}$. Then we compute the weighted sum of all the frame differences in $\mathbf{V}$, this is shown in Eq. (1).

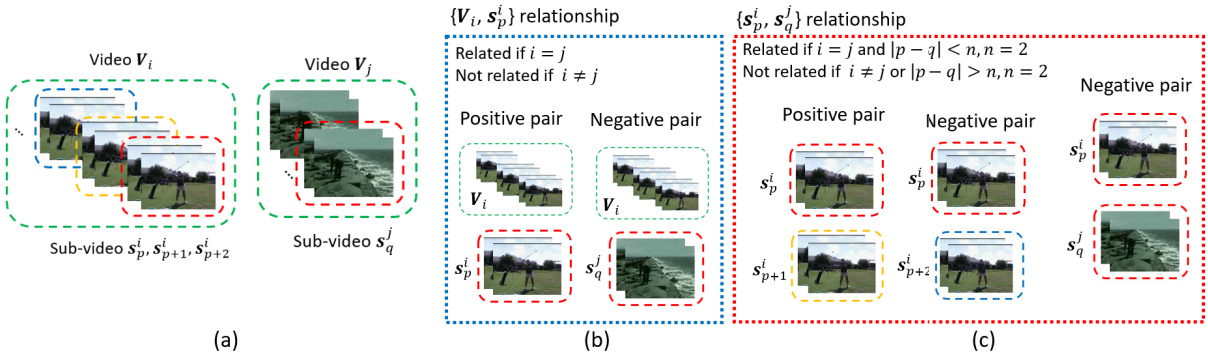

Fig. 1 Illustration of the learning scheme in our work. Best viewed in colors. (a) Two random videos $\mathbf{V}_i$ and $\mathbf{V}_j$, the colored dash boxes inside the videos indicate sub-videos. (b) The relationship between a video and a sub-video. (c) The relationship between two sub-videos. We will give more details in Sect. 2.2.

$$\mathbf{D} = \sum w_t \mathbf{d}_t, \ \textit{where } w_t = T + 1 - 2t \tag{1}$$

By adopting the sum of frame difference, the video $\mathbf{V} \in R^{h \times w \times 3 \times T}$ has been compressed to $\mathbf{D} \in R^{h \times w \times 3}$.

2.1.2 Dynamic Image Encoding

The core idea of dynamic image is to compress multiple video frames into one single image by considering the temporal order constraints. For a given video $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_T\}$, where $\mathbf{x}_t$ is the frame-level feature at time t. This can be done by modeling a ranking function as follows:

$$\mathbf{w}^* \in \arg\min_{\mathbf{w}} \left\{ \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{2} \sum_{t=1}^{J} [|t - \mathbf{w}^\top \mathbf{v}_t| - \epsilon]_{\geq 0}^2 \right\} \tag{2}$$

where $\mathbf{v}_t = \frac{1}{\|\mathbf{m}_t\|}\mathbf{m}_t$, and $\mathbf{m}_t = \frac{1}{t}\sum_{\tau=1}^{t} \mathbf{x}_\tau$, Eq. (2) can be optimized by [8]. The parameter $\mathbf{w}^*$ is the compressed video-level representation which has the same dimension as $\mathbf{x}_t$. If we set $\mathbf{x}_t$ as a vectorized video frame, then we could reshape the parameter $\mathbf{w}^*$ to the size of $h \times w \times 3$. For complete derivation of Dynamic image we refer the readers to [2].

2.1.3 Dynamic Optical Flow Encoding

This is similar to dynamic image encoding. We employ the iDT [9] to compute the optical flows, then normalize them to the range of $[0, 255]$. We compute the motion magnitude at each pixel then add this as a third channel to the optical flow. The dynamic optical flow can be computed using Eq. (2), and the size of dynamic optical flow is $h \times w \times 3$.

2.2 Network Architecture

We show our network architecture in Fig. 2. The network is trained to accomplish two different tasks: (a) whether a sub-video is related to a full video, and (b) whether two sub-videos are related to each other. More specifically, given two videos $\mathbf{V}_1 = \{\mathbf{s}_1^1, \ldots, \mathbf{s}_p^1\}$ and $\mathbf{V}_2 = \{\mathbf{s}_1^2, \ldots, \mathbf{s}_q^2\}$, where $\mathbf{s}_k^i$ is the k-th sub-video in video $\mathbf{V}_i$. We build the {*sub-video*, *video*} pairs as the input for task (a). We set a binary label

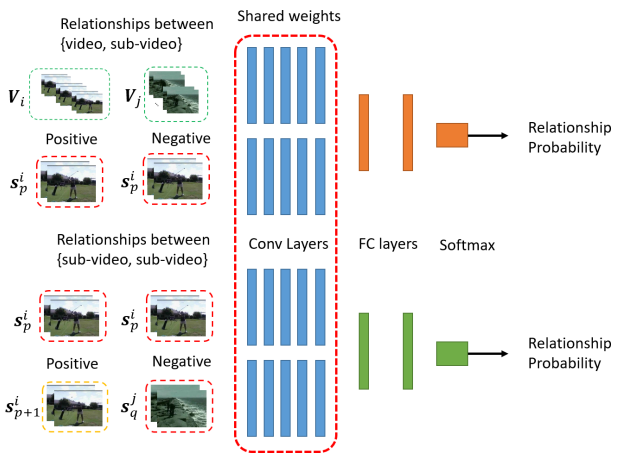

Fig. 2 The network architecture for {video, sub-video} relationship reasoning, and {sub-video, sub-video} relationship reasoning.

for the video pair, which is defined as:

$$y_R(\mathbf{V}_i, \mathbf{s}_k^j) = \begin{cases} 1, & \textit{if } i = j \\ 0, & \textit{otherwise} \end{cases} \tag{3}$$

For a {*sub-video*, *video*} pair, if they are from the same video, we define the pair as positive sample, otherwise, the pair is negative sample. Task (b) has a similar setting. We define the binary label for the video pair {*sub-video*, *sub-video*} as:

$$y_r(\mathbf{s}_p^i, \mathbf{s}_q^j) = \begin{cases} 1, & \textit{if } i = j \textit{ and } |p - q| < n, \ n = 2 \\ 0, & \textit{otherwise} \end{cases} \tag{4}$$

The only difference is we take the temporal relationship into consideration in task (b). To accomplish the two tasks, we employ the Siamese architecture [5] in our work. The network takes the encoded video pair as input. After the last pooling layer we concatenate the output from both streams. The fully-connected layer will compute a binary probability, which indicates the relationship of the video pair. We use the similar settings as [5]. The network can be trained by minimizing binary cross-entropy loss function. We jointly

train the two tasks by using a joint loss in a weighted sum, the learned convolutional layer weights are shared among all the four streams. Different from [5], we set the number of neurons in the fully connected layers as 4096/2048. During test, we use the feature from $FC7$ as the video representation. The learned representation can be used as a feature extractor in both action anticipation and action recognition.

2.3 Motion-Based Temporal Segmentation

Given a video $\mathbf{V} = \{I_1, \ldots, I_T\}$, we first compute the optical flow using iDT [9]. The motion magnitude at each pixel can be computed as $m(x, y) = \sqrt{(u(x, y)^2 + v(x, y)^2)}$, where $u(x, y)$ and $v(x, y)$ are the horizontal- and vertical-component of the optical flow at position (x, y). We compute the motion intensity as $M(I) = \sum_{(x,y)\in I} m(x, y)$. Then we segment the video according to the motion intensity: each sub-video should begin and end with a local minimum motion intensity. We use the segmented sub-videos to build video pairs and train the network, we call this *motion-segment pre-train*. Then we divide training videos into fixed-sized sub-videos, and build video pairs to train the network again, we call this *uniform-segment fine-tune*.

3. Experiments

3.1 Datasets

We evaluate our proposed approach on two challengine datesets: UCF101 [6] and HMDB51 [7]. UCF101 contains 13k clips, and are collected in 101 action categories. HMDB51 contains 6849 clips, in 51 action categories. We use the train/test splits provided by the author for both datasets.

3.2 Implementation Details

We use the training data from UCF101 and HMDB51 without any annotations. In the *motion-segment pre-train* stage, the batch size is 48, the learning rate starts at 10^{-3}, changes to its $\frac{1}{10}$ every 5K iterations, the process ends at 15K iterations. In the *uniform-segment fine-tune* stage, we use three temporal scales to generate $\{25, 15, 10\}$ sub-videos with equal length. The batch size is 48, the learning rate starts at 10^{-4}, changes to its $\frac{1}{10}$ every 3K iterations, the process ends at 12K iterations. Different frame encoding strategies are trained separately. For HMDB51, we use the model pre-trained on UCF101, then follow the same process as UCF101, the training ends at 10K and 9K in the *motion-segment pre-train* and *uniform-segment fine-tune* stages. We employ scale-jittering [10] in four spatial scales {240, 224, 192, 168}. For joint training, we set the weight of {*video*, *sub-video*} and {*sub-video*, *sub-video*} networks to 0.7 and 0.3. The self-supervised network is implemented using MatConvNet [11] toolbox. When testing, a video/sub-video is first compressed to an image by frame encoding, the image is used as input to the corresponding network. We use the output of $FC7$ as the representation of the video, we show how to utilize this representation in Sect. 3.3.

3.3 Baselines

To evaluate the learned representation, we set two different baselines. We treat any test video as a combination of sub-videos. To decide the category of a video, we follow two criteria: (1) if none of the sub-videos are in the same category, we choose the category with maximum probability. (2) if multiple sub-videos are in the same category, we choose the category which contains more sub-videos than the others.

Nearest Neighbor: For a test video, we first extract its feature using our network, then find its nearest neighbor from the training data using Euclidean distance.

FC: Inspired by [12], we replace the weights of the deep network with the trained weights, we only fine-tune the fully-connected layers while keep the convolutional layers fixed. We follow the settings in [12] to fine-tune the fully-connected layers. Note that we treat both video and sub-video as training data in the supervised baselines, all sub-videos from the same video share the same class label.

3.4 Comparison with State-of-the-Art

In this section, we compare the proposed self-supervised learning to the state-of-the-art approaches on action anticipation and action recognition. The results are shown in Table 1.

To better evaluate the representative power of each approach, we adopt all these approaches under baselines. FD, DI and DOF indicate frame difference, dynamic image and dynamic optical flow with *motion-segment pre-train*. CN indicates the representation concatenation of dynamic image and dynamic optical flow in our work. MS indicates motion-based temporal segmentation introduced in Sect. 3.2. The percentage in the second row indicates how much of a video we observed, in the case of 100%, this turns out to be a action recognition problem. [12] propose a self-supervised

Table 1 Average accuracies of different video frame encoding (splits 1).

		UCF101 [6]			HMDB51 [7]		
		10%	50%	100%	10%	50%	100%
NN	OOO [12]	6.5	18.6	32.5	5.3	11.5	18.6
	RkSm [4]	4.3	13.2	21.6	4.5	7.9	12.1
	FVis [1]	3.7	11.6	19.4	3.6	7.8	11.5
	FD	5.9	15.4	26.5	5.4	11.0	17.9
	DI	6.4	17.9	31.2	5.9	12.1	18.9
	DOF	5.8	16.6	29.4	5.2	11.4	18.5
	CN w/o MS	6.2	17.6	29.7	5.5	11.7	18.5
	CN w/ MS	7.0	18.9	32.5	6.7	12.4	20.2
FC	OOO [12]	11.5	33.5	**57.9**	7.9	18.4	31.4
	RkSm [4]	8.3	27.2	38.7	5.8	9.6	14.4
	FVis [1]	7.5	25.7	34.3	4.4	8.9	13.7
	FD	10.9	30.4	49.2	7.6	16.5	25.6
	DI	13.5	34.5	53.6	8.6	20.2	30.5
	DOF	12.7	33.1	52.5	8.0	19.7	28.4
	CN w/o MS	13.6	33.8	53.9	8.2	19.5	29.2
	CN w/ MS	**15.3**	**36.4**	56.3	**9.6**	**21.3**	**32.6**

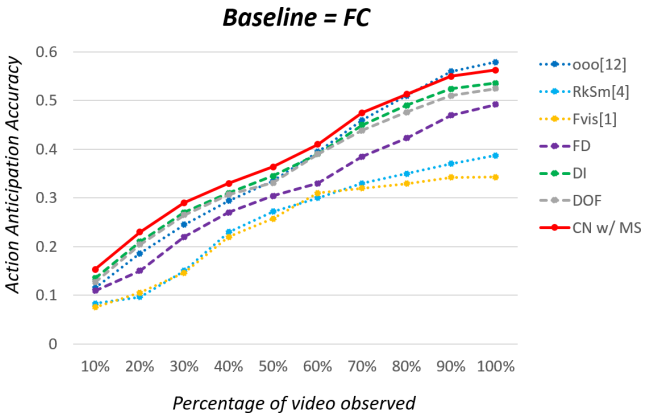

Fig. 3 Action anticipation on UCF101 dataset. Our representation can achieve better performance than the state-of-the-art self-supervised learning approaches.

framework, which uses the number of videos with correct order as supervisory signal. [1] designs a self-supervised network to predict the future visual representation of unlabeled videos. [4] uses the distance ranking of objects similarity in videos to train a self-supervised network. We implement these approaches in our work, [12] and [4] can be directly used as feature extractors. [1] in our work is also used as a feature extractor, the only difference is that it extracts the future visual representation, for action recognition, we do not use the full video as input for [1].

As shown in Table 1, our proposed representation can achieve better results than the state-of-the-art self-supervised learning approaches in action anticipation. We think this is mainly because we model the relationship between the videos and sub-videos and design the network to learn to reason about this relationship. And we can further improve the results if we combine dynamic image and dynamic optical flow together. We think this is because both appearance and motion are important cues in video process. The results also show that adding a *motion-segment pretrain* can help us to improve the final results. To show the efficiency of the proposed representation in action anticipation, we illustrate a more detailed results on UCF101 dataset in Fig. 3.

4. Conclusion

In this paper, we propose a self-supervised learning method which aims to anticipate actions from only partially observed videos. We use the abundantly available unlabeled videos to train the proposed network. In the learning process, the proposed network learns to reason about the relationships between videos/sub-videos and sub-videos, this will lead to better generalize the feature. The learned representation can then be used in both action anticipation and action recognition. Experiments show we could achieve comparable results with the state-of-the-art self-supervised learning approaches for both action anticipation and action recognition.

Acknowledgments

This work was supported in part by National Natural Science Foundation of China under Grant 61601102.

References

[1] C. Vondrick, H. Pirsiavash, and A. Torralba, "Anticipating visual representations from unlabeled video," 2016 IEEE Conference on Computer Vision and Pattern Recognition, pp.98–106, 2016.
[2] H. Bilen, B. Fernando, E. Gavves, A. Vedaldi, and S. Gould, "Dynamic image networks for action recognition," Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp.3034–3042, 2016.
[3] L.C. Pickup, Z. Pan, D. Wei, Y. Shih, C. Zhang, A. Zisserman, B. Scholkopf, and W.T. Freeman, "Seeing the arrow of time," IEEE Conference on Computer Vision and Pattern Recognition, pp.2043–2050, 2014.
[4] X. Wang and A. Gupta, "Unsupervised learning of visual representations using videos," 2015 IEEE International Conference on Computer Vision, pp.2794–2802, 2015.
[5] O. Sümer, T. Dencker, and B. Ommer, "Self-supervised learning of pose embeddings from spatiotemporal relations in videos," The IEEE International Conference on Computer Vision (ICCV), Oct. 2017.
[6] K. Soomro, A.R. Zamir, and M. Shah, "UCF101: A dataset of 101 human actions classes from videos in the wild," arXiv preprint arXiv:1212.0402, 2012.
[7] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "HMDB: A large video database for human motion recognition," Proc. International Conference on Computer Vision (ICCV), 2011.
[8] A.J. Smola and B. Schölkopf, "A tutorial on support vector regression," Statistics and Computing, vol.14, no.3, pp.199–222, 2004.
[9] H. Wang and C. Schmid, "Action recognition with improved trajectories," IEEE International Conference on Computer Vision, pp.3551–3558, Dec. 2013.
[10] 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," European Conference on Computer Vision, Lecture Notes in Computer Science, vol.9912, pp.20–36, Springer, 2016.
[11] A. Vedaldi and K. Lenc, "MatConvNet: Convolutional neural networks for MATLAB," Proc. 23rd ACM International Conference on Multimedia, pp.689–692, ACM, 2015.
[12] B. Fernando, H. Bilen, E. Gavves, and S. Gould, "Self-supervised video representation learning with odd-one-out networks," CVPR, pp.5729–5738, 2017.