Explore Video Clip Order with Self-Supervised and Curriculum Learning for Video Applications

Jun Xiao, Lin Li, Dejing Xu, Chengjiang Long, Jian Shao, Shifeng Zhang, Shiliang Pu, and Yueting Zhuang

***Abstract*—We present a self-supervised spatiotemporal learning approach by exploring the temporal coherence of videos. The chronological order of shuffled clips from the video is used as the supervisory signal to guide the 3D Convolutional Neural Networks (CNNs) to learn meaningful visual knowledge. Unlike the existing approaches which use frames, we utilize dynamic video clips to reduce the uncertainty of order. We test three types of representative 3D CNNs, all of which benefit from the proposed approach. The learned 3D CNNs can be used either as a feature extractor or a pre-trained model for further fine-tuning on downstream tasks. We also propose two curriculum learning strategies to make the 3D CNNs easier to train and get the state-of-the-art results in nearest neighbor retrieval and action recognition tasks compared with other self-supervised learning methods. Meanwhile, it is further extended to the field of visual question answering application and has achieved promising results. Besides, comprehensive and extensive experimental results and analyses are provided for readers to better understand the video clip order we explore with self-supervised and curriculum learning for video application.**

***Index Terms*—Self-supervised learning, curriculum learning, nearest neighbor retrieval, action recognition, video question answering.**

I. Introduction

In the field of computer vision, Convolutional Neural Networks (CNNs) [1], [2] have been in a hegemonic position recently. Nonetheless, the burgeoning of CNNs is mainly dependent on manually annotated large-scale datasets, such as ImageNet [3] and PASCAL VOC [4]. As a typical kind of CNNs, 3D CNNs have been explored primarily in action recognition [5]–[7] for a long time. In particular, many video applications such as action recognition [8], video retrieval [9] and video question answering [10], are always of great significance for their applicability. In the field of videos, due to the lack of similar large-scale manually annotated datasets, the parameters of 3D CNNs cannot be fully optimized like 2D CNNs, so that 2D CNNs which take both the RGB and flow streams [11] as inputs can still compete with 3D CNNs in action recognition. In [12], several successful image classification model structures are extended and trained on Kinetics [6] video dataset. The authors conclude that 3D CNNs and Kinetics may make significant advances in areas connected to various video tasks, just as 2D CNNs and ImageNet do.

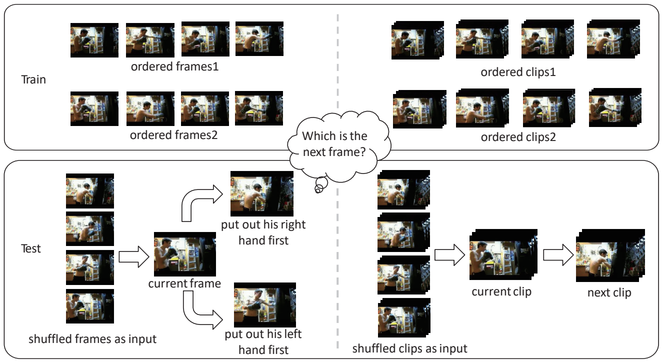


Fig. 1. Illustration of the necessity to use clips. In the training phase, there are corresponding samples for the different orders of punches, while in the test phase, the correct order is hard to predict if given only the frame. When clips are utilized for the order prediction task, such confusion can be solved well by using the dynamic information that the clip contains.

At present, although large-scale video datasets begin to sprout [6], [13], how to learn meaningful representations from unlabeled data is always a hot focus due to the heavy cost of annotation. A method called self-supervised learning is developed to adopt supervised machine learning techniques on unlabeled data by designing pretext tasks. For instance, there are self-supervised tasks such as predicting relative positions of image patches [14], solve jigsaw puzzles [15], image color channel prediction [16] and image inpainting [17]. For video data, since the particularity of temporal information exists, some recent works also attempt to leverage the temporal relationship among frames, such as order verification [18], [19] and order prediction [20] of frames.

The existing framework of self-supervised works tends to leverage the video in frame-level. The features of frames are extracted by 2D CNNs, then integrated to predict the verification results or the actual order of the input frames. The trained CNNs can be used either as a feature extractor or a pre-trained model for fine-tuning on classification and detection. Compared with order verification [18], [19], order prediction [20] contains richer supervisory signals and indicates better performance in several validation experiments. Admittedly, order prediction with frames sometimes contains ambiguity, as shown in Fig. 1. During the training, there are two different kinds of ordered frames. The top is the boxer hitting the

This work was supported by the National Key Research and Development Project of China (No. 2018AAA0101900), the National Natural Science Foundation of China (U19B2043, 61976185), Zhejiang Natural Science Foundation (LR19F020002, LZ17F020001), the Fundamental Research Funds for the Central Universities, Chinese Knowledge Center for Engineering Sciences and Technology, and Joint Research Program of ZJU & Hikvision Research Institute. *(Corresponding author: Dejing Xu.)*

J. Xiao, L. Li, D. Xu, J. Shao and Y. Zhuang are with Zhejiang University, Hangzhou, 310000, China (e-mail: junx@cs.zju.edu.cn; mukti@zju.edu.cn; xudejing@zju.edu.cn; jshao@cs.zju.edu.cn; yzhuang@zju.edu.cn).

C. Long is with Computer Vision Team at Kitware Inc, New York, 10001, USA, (e-mail: cjfykx@gmail.com).

S. Zhang, S. Pu are with Hikvision Research Institute, Hangzhou, 310000, China (e-mail: zhangshifeng@hikvision.com; pushiliang.hri@hikvision.com).

sandbag with his left fist first, and the bottom is the opposite, which will make the model confusing about the task. To conquer this deficiency as far as possible, [20] takes forward and backward orders as the same category. It is a compromise under the circumstances that merely frames and 2D CNNs are applied.

By contrast, we recommend using clips and 3D CNNs directly to make the task more explicit. Because each clip contains internal dynamics, the order will be more distinguishable if given a shuffled sequence of clips. We integrate 3D CNNs into the clip order prediction task. Firstly, several clips of fixed length and interval are randomly sampled from the video for shuffling, then 3D CNNs are utilized to extract the features of these clips, and ultimately, these features are fed into a simple neural network to predict the actual order of the shuffled clips. These optimized 3D CNNs can learn visual prior knowledge via clip order prediction tasks, which can be applied to other video-related tasks for better performance.

We shall point out that the foundation of the proposed approach in this paper is firstly published in our previous work [21]. In the previous work, when the number of clips per tuple increases, the entire framework exhibits turbulent training and is hard to converge. This paper extends the initial paper by adopting two curriculum learning strategies to settle the matter, and enumerating more experimental comparisons and analyses to better understand the proposed self-supervised learning method for video applications involving nearest neighbor retrieval, action recognition, and video question answering. To summarize, the main contributions of the paper are as follows:

- We propose to use video clips in order prediction, which is more consistent with video dynamics and enables the self-supervised spatiotemporal learning of 3D CNNs.
- Two curriculum learning strategies are proposed to ease the training of the model. We also give detailed analyses of the clip order prediction results to better understand the proposed task.
- By experimenting with C3D, R3D and R(2+1)D networks under diverse task settings, we prove that the proposed method has wide applicability.
- We evaluate the learned 3D CNNs in nearest neighbor retrieval and action recognition tasks, and we get state-of-the-art performances in both tasks, which also shows the effectiveness of the proposed method. In addition, we further evaluate the efficiency of trained models as feature extractors on video question answering task.

The rest of the paper is organized as follows. We first review related works in Section II, then we explain the details of the proposed method in Section III. In Section IV, the implementation and results of the experiments are provided and analyzed. Finally, we conclude our works in Section V.

II. Related Work

In this section, we briefly introduce recent researches of action recognition, self-supervised learning, and curriculum learning that related to our work.

A. Action Recognition

As one of the classical problems in the field of computer vision, the basic pipeline of action recognition is to extract features first, then classify them. Nowadays, there are three primary methods for action recognition, which are the traditional methods [22]–[27], 2D CNNs-based methods, and 3D CNNs-based methods respectively.

Since AlexNet [1] made a breakthrough in image classification, researches that use 2D CNNs in action recognition tasks have emerged [11], [28], [29]. In [11], the input video is decomposed into the spatial flow and the optical flow. The deep 2D CNNs are utilized to process each stream, and the action categories are predicted through the later fusion. [29] proposes three fusion methods to integrate the temporal information of the video. It also implements multiresolution by dividing the input frames into context streams and fovea stream. Some other improved models based on the two-stream model are also used for action recognition, such as trajectory-pooled deep convolutional descriptors (TDD) [30], temporal segment network (TSN) [31], etc.

The 3D CNNs [5], [32], [33] extend 2D CNNs to the temporal domain and extract spatiotemporal features for action recognition. In [6], 2D CNNs trained on ImageNet are converted to 3D CNNs by inflating all filters and pooling kernels. [5] proposes the C3D network in which 3D convolution kernels are stacked and followed by fully connected layers. In recent studies, ResNet [2] architecture is also extended from 2D convolution kernel to 3D convolution kernel. [7] proposes a novel ResNet structure called P3D ResNet, which constructs three types of bottleneck building blocks and interleaves them. The decomposition of 3D convolution to 2D spatial and 1D temporal convolutions are proposed in [33]. In [12], they focus on training very deep 3D CNNs from scratch and point out that training deeper 3D CNNs on large datasets is more effective.

B. Self-Supervised Learning

Self-supervised learning is a technique for learning representations or priors by completing a specific surrogate task, in which supervisory signals are generated automatically. It provides a compelling way to leverage abundant unlabeled data. In self-supervised learning, data structures are designed to generate diverse pretext tasks. The learned model from pretext tasks can be directly reused on downstream tasks either for feature extraction or for fine-tuning. Because of this characteristic, self-supervised learning is widely used in computer vision tasks for the numerous unlabeled images and videos.

For the proxy task that uses images, the representation of the image is prevailingly learned by restoring the spatial information. For instance, [14] proposes to learn image representation by predicting the relative position between two image patches. These patches are sampled from the same image in eight spatial arrangements. In [15], nine tiles are extracted from the image and shuffled according to a predefined permutation set to make jigsaw puzzles. Based on the hamming distance between all possible permutations, the permutation set is determined by a greedy algorithm. [34] proposes to learn

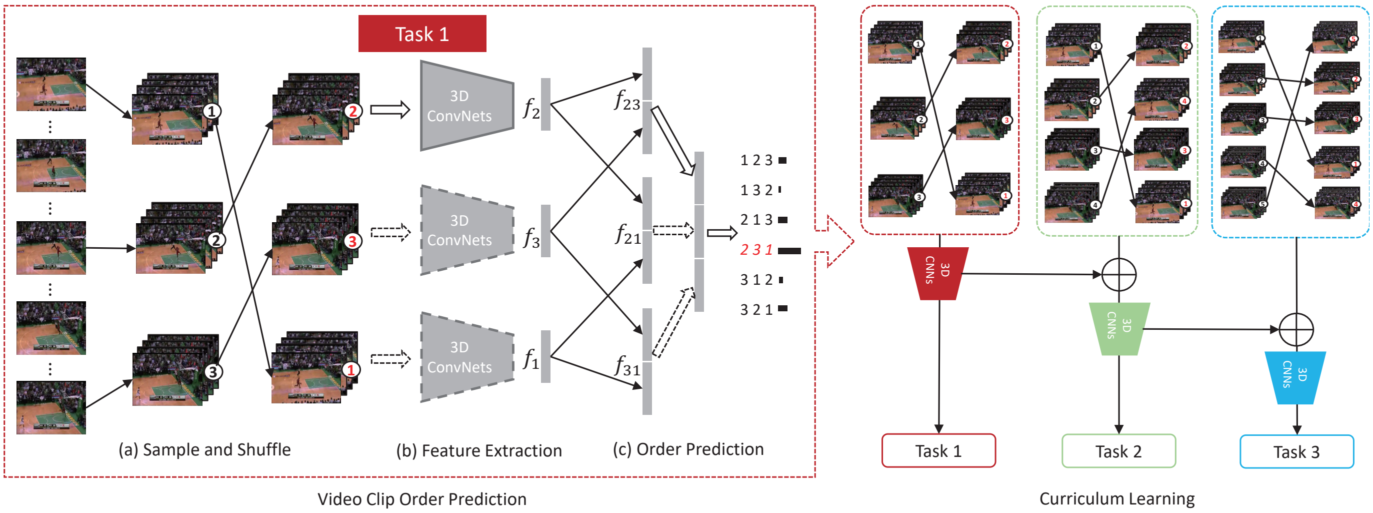


Fig. 2. Left: the overview of clip order prediction framework. (a) sample and shuffle: evenly sample non-overlapping clips and randomly shuffle them. (b) feature extraction: use 3D CNNs to extract the features of all clips. 3D CNNs are randomly initialized. (c) order prediction: the extracted features are pairwise concatenated and transformed, then concatenated together again and fed into a fully-connected layer to predict the actual order. The dotted lines indicate that the corresponding weights are shared. Right: the overall process of task-based curriculum learning. As the number of clips per tuple increase from 3 to 5, the difficulty of the task changes from easy to hard.

visual features by training 2D CNNs to recognize the rotation degree of images. In addition, [17] exploits an encoder-decoder architecture to tackle the image inpainting task, while [16] employs image colorization as the pretext task to learn semantic features of images.

Compared with image, video is temporal-coherent and dynamic, thus most studies regard the temporal information in the video as the supervisory information. [35] utilizes transitive relations to learn representations invariant to inter-instance and intra-instance variations among object patches. These object patches are extracted from unlabeled videos via motion cues. In [36], ranking machines are utilized to capture the evolution of appearances among the frames, and the learned functional parameters can be used as the video representation. In [18], video representation is learned by judging whether the frame sequence is ordered. [19] proposes to the odd-one-out network to identify odd elements in a group of related elements. [20] proposes another related task in which the actual order of input frames should be predicted. The task is formulated as a multi-category classification problem, with forward and backward orders grouped into the same category. Since the number of possible permutations or orders explodes as patches or frames are added, permutations are always predefined, as mentioned earlier in [15]. While in [37], they propose a reinforcement learning algorithm, which exploits 2D CNNs to predict the spatial and temporal order of frames, and updates training permutations according to the network states adaptively.

Most of the above studies use frames as the input to complete the proxy task of video, thus the learned CNNs are purely capable of extracting features for still images. In order to better leverage the strength of 3D CNNs and the internal dynamics of videos, we extend the order prediction task [20] from frames to clips in our previous work. Recent studies have also attempted to take advantage of the dynamics of videos. In [38], they propose a motion and appearance statistics prediction task to capture high-level concepts of videos. [39] expands the jigsaw task from 2D to 3D, so as to learn intricate spatiotemporal video representation. [40] proposes to learn representations by predicting the transformations applied to the current clip given its surrounding ones, while [41], [42] utilize the video speed recognition as a proxy task. In this paper, we optimize our approach in more complex task settings to enable the model to learn richer spatiotemporal information and internal dynamics.

C. *Curriculum Learning*

Aiming at training complex network in deep learning, Bengio et al. [43] proposes curriculum learning. The main idea is to imitate the characteristics of human learning, from simple to hard to learn the curriculum gradually, so that model can perceive a better local optimum and accelerate training simultaneously. [44] proposes an algorithm that can automatically discover the favorable sequence of tasks. [45] applies curriculum learning to neural machine translation by rearranging the order of samples according to similarity scores. [46] proposes a novel framework that integrates the original curriculum learning with the self-paced learning [47].

In our previous work [21], we indicate that as the number of clips per tuple increases, the complexity of the prediction task overgrows in a factorial level. And at the same time, the model converges slowly during the training, while a higher learning rate will fluctuate the training process. In this work, we apply the curriculum learning in two ways, which considers both the sample and task difficulty. The concrete details will be explained in the next section.

III. Clip Order Prediction

In this section, we will begin with a brief overview of the proposed clip order prediction method, then describe each part of it in detail.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2020.3025661, IEEE Transactions on Multimedia

The clip order prediction task mainly consists of three processes: sample and shuffle, feature extraction and order prediction. The left of Fig. 2 shows the overall framework. In the process of sample and shuffle, we evenly sample and shuffle multiple clips. In feature extraction, several 3D CNNs with the shared weights are used to extract the features of the clips separately. In the process of order prediction, we follow the classification model proposed in [20]. The extracted features are pairwise concatenated and forwarded through two linear layers, and finally through the softmax operation to obtain the probability distribution on each permutation.

Next, we introduce several definitions of clip order prediction proxy task, then elaborate on the three processes mentioned above. A clip is composed of continuous frames with a uniform sampling size $c \times l \times h \times w$ from the video, where c, l, h and w denotes the number of channels, clip length, height and width of each frame respectively. The size of the 3D convolution kernel is $t \times d \times d$, where t is the temporal depth and d is the spatial size. We define an ordered clip tuple as $\boldsymbol{C} = \langle c_1, c_2, \ldots, c_n \rangle$, and the features extracted by 3D CNNs are expressed as $\boldsymbol{F} = \langle \boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_n \rangle$. The subscripts here represent the chronological order.

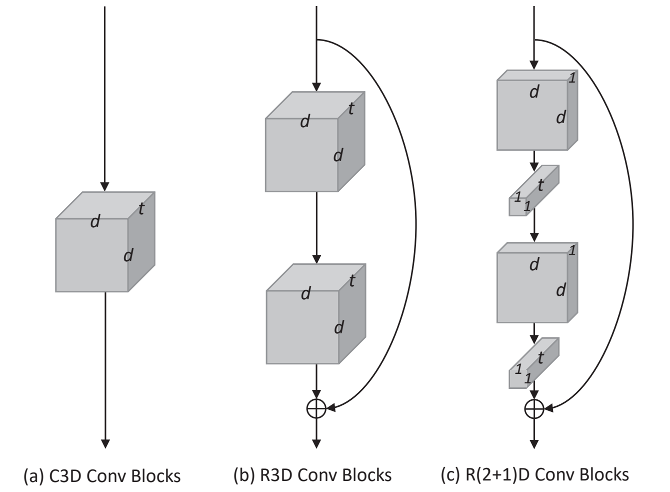
A. Sample and Shuffle

In this process, we randomly sample consecutive frames, namely clips, from the video. For N clips, the possible permutation set encompasses $N!$ elements. Taking N = 7, the aggregate of possible permutations will be 7! = 5040. The difficulty of classification task surges when the number of clips per tuple increases. Previous works [15], [37] select a few specific orders from all possible orders during the training. Since clip order prediction is purely a proxy task, and we focus on the learning of 3D CNNs, this task should be solvable. Otherwise, if the entire proxy task is too complex, it is hard to learn a desirable video representation. Therefore, we limit the number of clips to between 2 and 5, and the maximum number of elements in the permutation set is just 120, which greatly reduces the training difficulty of the model.

Considering an extreme case where two clips are overlapped by 1 frame, then the order task can be settled by simply comparing the pixels of the frames. To avoid such a situation where the whole framework handles the task by comparing low-level characteristics like texture and color, we sample clips from the video evenly spaced by m frames. After sampling, the clips are shuffled to form the input. The shuffle step is forced to be random, no particular permutations are preferred. In the training phase, the number of generated samples belonging to diverse order categories is roughly the same.

B. Feature Extraction

Three kinds of 3D CNNs, C3D [5], R3D [33] and R(2+1)D [33], are used to extract features from previous shuffled clips. The structures of various convolutional blocks are exhibited in Fig. 3. The same 3D CNNs are used for all clips in one tuple, as Fig. 2 (b) shows. We will illustrate the architecture of each network in detail below.



(a) C3D Conv Blocks (b) R3D Conv Blocks (c) R(2+1)D Conv Blocks

Fig. 3. Three kinds of 3D conv blocks. (a) C3D Conv Blocks: the classic 3D convolution kernel with size $t \times d \times d$, which are stacked to form the C3D network. (a) R3D Conv Blocks: classic 3D convolution kernels with a shortcut connection. (c) R(2+1)D Conv Blocks: the 3D kernel are decomposed into a spatial 2D kernel $(1 \times d \times d)$ and a temporal 1D kernel $(t \times 1 \times 1)$. Batch normalization and ReLU layers are omitted for clarity.

1) C3D: The model is an extension of 2D CNNs with a temporal dimension. 3D CNNs extract both temporal and spatial dimensions via 3D convolution, and are capable of modeling the dynamics of video, which is well-suited for spatiotemporal learning [5], [48]. The C3D network consists of 8 successively stacked convolutional layers, 5 staggered pooling layers, and followed by two fully connected layers terminally. In [5],the author concludes that the homogeneous setting of $3 \times 3 \times 3$ convolution kernel is the best practice.

2) R3D: Residual learning principle [2] is a milestone for the architecture design of 2D CNNs. To effectively train the deep network and solve the network degradation problem, the bypass mechanism is introduced in ResNet. It prompts the performance of many image-related tasks, such as classification, detection, and segmentation to the state-of-the-art. R3D uses a similar design in 3D convolution, and it can be used for video processing to perceive spatiotemporal information. The operations of the basic convolutional block are as follows:

$$\boldsymbol{x}_o = \mathcal{F}_2(\mathcal{F}_1(\boldsymbol{x}_i)) + \mathcal{H}(\boldsymbol{x}_i) \tag{1}$$

Where $\boldsymbol{x}_i$ and $\boldsymbol{x}_o$ represent the input and output of the block respectively, $\mathcal{F}$ stands for 3D convolution operation, and $\mathcal{H}$ is a function that scales the $\boldsymbol{x}_i$ to the size of $\boldsymbol{x}_o$ when necessary. The convolution block is composed of two 3D kernels, with batch normalization and ReLU layers appended. There are 5 convolutional layers in total, and the specification can be referred in Table 1 of [33].

3) R(2+1)D: 3D convolution can be decomposed into two separate and successive operations, one is 2D spatial convolution, the other is 1D temporal convolution. The procedure can be refactored by first applying spatial convolution then temporal convolution. The specific operations of the convolution block are as follows:

$$\begin{aligned} \boldsymbol{x}_m &= \mathcal{T}_1(\mathcal{S}_1(\boldsymbol{x}_i)) \\ \boldsymbol{x}_o &= \mathcal{T}_2(\mathcal{S}_2(\boldsymbol{x}_m)) + \mathcal{H}(\boldsymbol{x}_i) \end{aligned} \tag{2}$$

Where $\boldsymbol{x}_i$, $\boldsymbol{x}_m$ and $\boldsymbol{x}_o$ correspond to the input, middle, and output of the block, respectively, and $\mathcal{S}$ represents the spatial convolution, $\mathcal{T}$ represents the temporal convolution, and $\mathcal{H}$ is the same as the previously mentioned function. The overall architecture is the same as R3D, except that more nonlinear layers like ReLU are inserted into the block, which means that the number of nonlinearities is doubled while the number of parameters to be optimized is almost the same.

Recent research indicates that the R(2+1)D network can achieve the state-of-art results on four action recognition benchmarks [33]. Both R3D and R(2+1)D networks employ the global spatiotemporal pooling layer to aggregate the activations after the convolutional layers. The gained vector is viewed as the semantic feature of the input clip. We modify the original C3D network to follow a similar design to the other two network structures.

C. Order Prediction

The order prediction is formulated as a classification task with a tuple of clip features as input and the possibility distribution of various orders as output. Experiments certify that the network in [20] has significant effects on both order prediction and learning of the underlying feature extractors. Therefore, we adopt the same method by leveraging a simple multi-layer perceptron, and the extracted features are pairwise concatenated at first. Given the extracted features, the operations are as follows:

$$\begin{aligned} \boldsymbol{h}_k &= g_\theta(\boldsymbol{W}_1(\boldsymbol{f}_i \| \boldsymbol{f}_j) + \boldsymbol{b}_1) \\ \boldsymbol{a} &= \boldsymbol{W}_2 \|_{k=1}^{N} \boldsymbol{h}_k + \boldsymbol{b}_2 \\ p_i &= \frac{\exp(a_i)}{\sum_{j=1}^{C} \exp(a_j)} \end{aligned} \tag{3}$$

Where $\|$ means the concatenation of vectors, g_θ is a nonlinear function, $\boldsymbol{W}$ and $\boldsymbol{b}$ are the parameters of linear transformation, $\boldsymbol{h}_k$ captures the relationship between $\boldsymbol{f}_i$ and $\boldsymbol{f}_j$, $\boldsymbol{a}$ is the logits, and p_i is the probability that the order belongs to class i.

Suppose a tuple contains 3 clips, and clips $\boldsymbol{C} = \langle c_2, c_3, c_1 \rangle$ are obtained via shuffling, and corresponding features $\boldsymbol{F} = \langle \boldsymbol{f}_2, \boldsymbol{f}_3, \boldsymbol{f}_1 \rangle$ are extracted. As shown in Fig. 2 (c), the extracted features are firstly pairwise concatenated as $\langle \boldsymbol{f}_{23}, \boldsymbol{f}_{21}, \boldsymbol{f}_{31} \rangle$, and then transformed into tuples of 3 vectors to capture the relationship among each clip. These vectors are concatenated again and fed into the full-connected layer, and finally through the softmax function to output the probability of each order. The sum of the probabilities is 1 and the order corresponding to the maximum value is the prediction. The target classes are permutations of $\langle 1, 2, 3 \rangle$, one of which is the actual order $\langle 2, 3, 1 \rangle$.

The cross-entropy loss is used to calculate the deviation between the target and the predicted value. The formula is defined as follows:

$$\mathcal{L} = -\sum_{i=1}^{C} y_i log(p_i) \tag{4}$$

Where y_i and p_i represent the possibility that the sample belongs to the class i order in the groundtruth and prediction respectively, and C represents the number of all possible orders. After the calculation, the loss $\mathcal{L}$ is backpropagated, and the SGD optimizer is used to optimize the model parameters.

D. Curriculum Learning

1) Task-based: The right of Fig. 2 shows the overall process of task-based curriculum learning. As the number of clips per tuple increases, the complexity level of the task promotes. With reference to the idea of curriculum learning, we define $\mathcal{L}_3$, $\mathcal{L}_4$ and $\mathcal{L}_5$ as cost functions with the number of the clips per tuple is 3, 4, and 5, separately, while the clip length and interval length are fixed. At first, we optimize a relatively easy target $\mathcal{L}_3$. Since the task is simply a classification task with 6 classes, the model converges easily. In this case, the parameters θ of the model are trained to complete the clip prediction task corresponding to the 3 clips per tuple and obtain the minimum value at $\mathcal{L}_3$. Then we gradually increase the number of clips per tuple, that is, raise the difficulty of the training, while maintaining the optimal parameters of $\mathcal{L}_{tl}$. Here $\mathcal{L}_3$ is the highly smoothed version of the $\mathcal{L}_{tl}$. Since the model has been able to arrange 3 clips, add another clip is only a slight change, and θ can be gradually updated to reach the minimum value of $\mathcal{L}_4$. Similarly, based on the model of 4 clips per tuple, the optimization of the model of 5 clips per tuple is carried out, and θ eventually reaches the minimum value of $\mathcal{L}_5$. Initially, the model parameters tend to learn from tasks of shorter tuple length. The next training procedure involves minor changes in model parameters. With the continued training of the model, the parameters gradually adapt to the evolution from simple to complex tasks and enable the model to accomplish more complex tasks in the end.

2) Sample-based: For a convergent model, it still has the bias to sort specific kinds of videos better. But actually, all videos contain different dynamics that are useful for spatiotemporal learning. We propose a simple and effective strategy to help the model learn more visual knowledge from the existing data. During the training process, when the model converges, if the softmax value of the correct order is lower than the specific threshold t, we determine that it is hard for the model to order the shuffled clip sequence. In the subsequent training, we increase the proportion of these hard samples by removing other simple ones. By continuously training on hard examples, the model can learn more comprehensive visual priors.

IV. EXPERIMENTS

In the section, we first describe the implementation details and analyze the results of clip order prediction experiments concretely, then evaluate the learned 3D CNNs via nearest neighbor retrieval, action recognition and video question answering tasks.

A. Video Clip Order Prediction

Although the purpose of the proposed self-supervised method is to learn visual knowledge from unlabeled videos, we choose the experiment on UCF101 [49] dataset without labels because of its diversity and wide usage. UCF101 is an action recognition dataset of realistic action videos, collected from YouTube, having 101 action categories. With 13,320 videos from 101 action categories, UCF101 has large diversity regarding actions and with the presence of large variations in camera motion, object appearance and pose, object scale, viewpoint, cluttered background, illumination conditions, etc.

We utilize the currently popular PyTorch [50] deep learning library to implement the entire framework. Unlike the original C3D network, the C3D network used in this article is modified by replacing the two fully connected layers with the global spatiotemporal pooling layer, which is also utilized in the R3D network. We adjust the R3D network by applying non-repeating blocks in conv{2-5}_x to obtain a total of 9 convolutional layers. The R(2+1)D network follows the same architecture as the R3D network, with only the 3D kernel decomposed. The nonlinear layers we use are ReLU. In addition, the dropout regularization layers are applied between the fully-connected layers with p = 0.5 to against overfitting.

To express the experiment clearly, we use cl, it, tl to represent the clip length, interval length, and the number of the clips per tuple, and the basic unit of cl and it is frame. The previously mentioned self-supervised learning method is trained and tested on split 1 of UCF101. On-the-fly data augmentation is applied to prepare the input data. We randomly split 800 videos from the training set to do validation during training. The input video clips are first resized to 128 × 171, and then randomly cropped to 112 × 112 in the training. During validation or testing, the clip is cropped to the center. Since 3D CNNs [5], [12], [33] universally demand a 16-frames clip as input, we also choose cl = 16 frames. To avoid trivial solutions for the task, we assign it = 8 and 16 frames respectively. In order to make a comprehensive comparison of the effects of tuple length, we experiment with tl = 3, 4 and 5 clips.

Mini-batch stochastic gradient descent is utilized to optimize the model parameters. Memory consumption has always been an obstacle in large batches training of neural networks, especially for 3D CNNs. Recently [51] shows that small mini-batch size provides more up-to-date gradient calculations and yields more stable and reliable training. Thus we adopt small batches of 8 tuples for training. The learning rate, momentum and weight decay are set to 0.001, 0.9 and 0.0005, respectively. The training process contains 300 epochs, and the best model with the lowest validation loss is saved for further analysis.

It is inevitable that the complexity of classification task raises as tl increases, which is also verified in our previous work [21] that training from scratch causes unstable training when tl is larger than 3. Therefore, we adopt the task-based curriculum learning strategy stated before. Initially, the network is trained with tl = 3, and when tl increases, the network is fine-tuned based on the network trained with the previous tl. The network that has performed a simpler order task lays down a solid foundation for the optimization of a harder one. The comparison of accuracy on the validation set between C3D trained from scratch and C3D with task-based curriculum learning is displayed in Fig. 4, where cl = 16, it = 8 and tl = 4. As we can see, randomly initialized C3D makes merely a puny growth in accuracy before the 200 epoch. Even if the maximum is achieved after 200 epochs, it cannot compete with the minimum of C3D (task-based) even in 10 epochs. It can be seen that C3D (task-based) has a faster convergence rate than randomly initialized C3D. For the following experiments, we will use the task-based training strategy for tl larger than 3 if not specified.

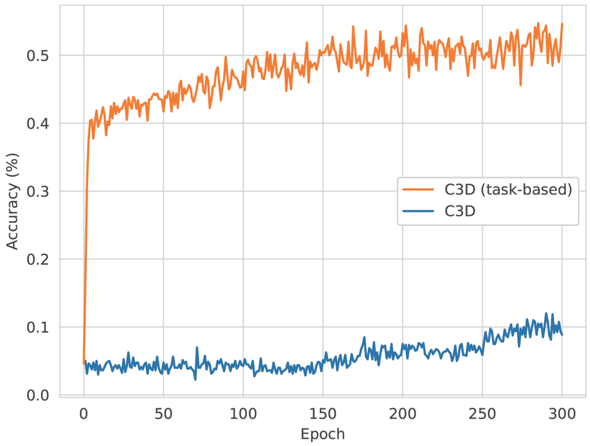


Fig. 4. The accuracy in the training process of the C3D network on the validation set when the number of clips per tuple is 4. Note that C3D is trained end-to-end from scratch, while C3D (task-based) is trained under task-based curriculum learning.

TABLE I
CLIP ORDER PREDICTION RESULTS ON UCF101. C3D, R3D AND R(2+1)D NETWORKS ARE TRAINED WITH CLIP ORDER PREDICTION FRAMEWORK SEPARATELY. * MEANS THAT THE SAMPLE-BASED CURRICULUM LEARNING STRATEGY IS APPLIED.

Model	cl	it	tl	Accuracy
C3D	16	8 (16)	3	68.5 (57.9)
	16	8 (16)	4	49.5 (65.4)
	16	8 (16)	5	23.2 (40.4)
R3D	16	8 (16)	3	68.4 (20.6)
	16	8 (16)	4	52.2 (10.6)
	16	8 (16)	5	28.2 (2.7)
R3D*	16	8 (-)	3	**70.1** (-)
R(2+1)D	16	8 (16)	3	64.2 (46.7)
	16	8 (16)	4	50.2 (70.8)
	16	8 (16)	5	27.9 (51.9)

The results of C3D, R3D and R(2+1)D on the clip prediction task under variant cl, it and tl conditions are exhibited in Table I. The task-based training is utilized for both it = 8 and 16 separately. Considering that the accuracies of random guessing for these tasks are 16.7%, 4.2% and 0.8% corresponding to tl of 3, 4 and 5, the framework indeed learns to analyze the content of clips and reason the order out. For it = 8, when tl increases, the accuracy decreases for all of the three models, which is reasonable since the difficulty of the task grows quickly. But there are also abnormal behaviors when it increases. For C3D and R(2+1)D model, when it = 16, though the task complexity grows from tl = 3 to 4, the order accuracy is also improved apparently. While under the same situation, the accuracy decreases for R3D. In our experiment,

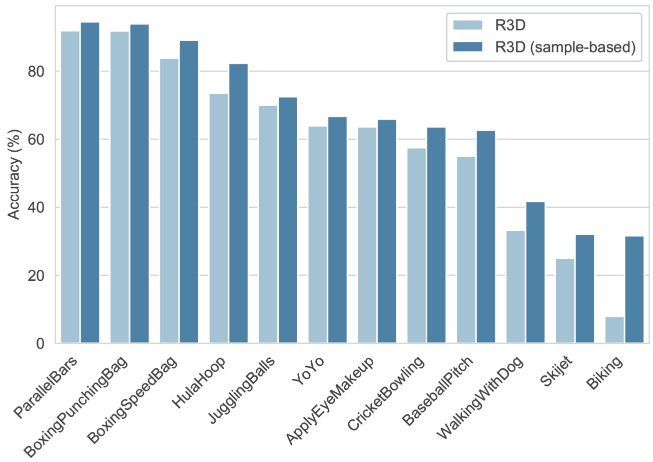


Fig. 5. The comparison of clip order prediction accuracy in different categories on UCF101 for R3D and R3D (sample-based), where cl = 16, it = 8, tl = 3.

the R3D model can get an accuracy of 20.6% in 300 epochs, and if trained with another 300 epochs, its accuracy can raise to 29.2% and still seems to improve. This means when tl = 4, the R3D converges very slowly compared to C3D and R(2+1)D, which may result from the architecture design.

For sample-based curriculum learning, we experiment with the R3D network. To reduce the impact of randomness, we collect the softmax value of the predictions in several convergent epochs to measure the difficulty of the corresponding video. If every classification is correct and the softmax value of the correct order is greater than the threshold 0.8 in these epochs, we will screen out such simple samples in the following training process. The order prediction accuracy of the model trained in this way is displayed as R3D* in Table I. Obviously, the accuracy has been improved further.

In addition, since the dataset we use has labels, we also calculate the average order prediction accuracy by each category on split 1 of UCF101, and several categories are displayed in Fig. 5. We discern that some categories that contain quick and repetitive actions, juggling balls, boxing speed bag, hula hoop and so on, all have an order accuracy of more than 70%, while it seems hard to arrange these disordered clips by human. We speculate that our trained 3D CNNs may have learned some video priors that are not visually accessible to human, such as flow direction, camera locations, etc. Meanwhile, we also perceive that there are some common characteristics on several categories with low accuracies, such as biking, skijet and walking with dog. In this way, the movement trend of each clip is likely to occur, making it difficult to attain the correct prediction. The sample-based training can enforce the model to spend more time learning from these hard samples and get higher accuracy correspondingly.

B. Nearest Neighbor Retrieval

As mentioned above, to complete the task, 3D CNNs inevitably analyze and understand the content of clips. As a feature extractor, the 3D CNNs is trained together with the whole framework. To facilitate comparison, we choose the nearest neighbor retrieval experiment used in [18], [37] to evaluate the quality of the learned representation. Besides UCF101, here we also utilize the HMDB51 [52] dataset as a comparison and supplement. HMDB51 is collected from various sources, mostly from movies, and a small proportion from public databases such as the Prelinger archive, YouTube and Google videos. The dataset contains 6,849 clips divided into 51 action categories, each containing a minimum of 101 clips.

TABLE II
FRAME AND CLIP RETRIEVAL RESULTS ON UCF101. THE TOP ROWS ARE BASED ON 2D CNNS, WHILE THE BOTTOM ROWS ARE BASED ON 3D CNNS. * MEANS THAT THE SAMPLE-BASED CURRICULUM LEARNING STRATEGY IS APPLIED.

Method	Top1	Top5	Top10	Top20	Top50
Jigsaw [20]	19.7	28.5	33.5	40.0	49.4
OPN [15]	19.9	28.7	34.0	40.6	51.6
Büchler et al. [37]	25.7	36.2	42.2	49.2	59.5
SpeedNet [42]	13.0	28.1	37.5	49.5	65.0
C3D (random)	16.0	22.5	26.7	31.4	39.3
C3D (16-8-3)	12.5	29.0	39.0	50.6	66.9
C3D (16-8-4)	15.3	32.5	42.7	53.8	69.7
C3D (16-8-5)	15.2	31.8	41.7	53.2	69.4
R3D (random)	10.5	17.2	21.5	27.0	36.7
R3D (16-8-3)	14.1	30.3	40.0	51.1	66.5
R3D (16-8-3)*	*15.0*	*31.9*	*41.7*	*53.1*	*68.5*
R3D (16-8-4)	**17.8**	**35.0**	**44.6**	**55.3**	**70.0**
R3D (16-8-5)	16.7	33.6	43.3	53.9	68.9
R(2+1)D (random)	10.2	17.3	21.9	27.7	38.5
R(2+1)D (16-8-3)	10.7	25.9	35.4	47.3	63.9
R(2+1)D (16-8-4)	14.1	31.1	40.8	52.0	67.5
R(2+1)D (16-8-5)	16.5	33.7	43.5	54.4	69.1

TABLE III
CLIP RETRIEVAL RESULTS ON HMDB51 WITH THE PROPOSED SELF-SUPERVISED LEARNING METHOD. * MEANS THAT THE SAMPLE-BASED CURRICULUM LEARNING STRATEGY IS APPLIED.

Model	Top1	Top5	Top10	Top20	Top50
C3D (random)	7.7	12.5	17.3	24.1	37.8
C3D (16-8-3)	7.4	22.6	34.4	48.5	70.1
C3D (16-8-4)	8.2	23.6	**35.7**	50.3	**70.9**
C3D (16-8-5)	7.2	21.6	32.9	47.6	69.5
R3D (random)	5.5	11.3	16.5	23.8	37.2
R3D (16-8-3)	7.6	22.9	34.4	48.8	68.9
R3D (16-8-3)*	*7.8*	*23.3*	*35.2*	*49.1*	*69.5*
R3D (16-8-4)	**8.9**	**24.2**	**35.7**	**50.4**	70.5
R3D (16-8-5)	8.7	23.2	34.9	49.1	69.0
R(2+1)D (random)	4.6	11.1	16.3	23.9	39.3
R(2+1)D (16-8-3)	5.7	19.5	30.7	45.8	67.0
R(2+1)D (16-8-4)	7.8	22.3	34.1	48.7	68.9
R(2+1)D (16-8-5)	8.3	23.2	35.0	49.4	69.7

In the nearest neighbor retrieval experiment of [37], they extract 10 frames per video and choose the pool5 layer of CaffeNet [53] as the representation. We follow the experimental setup in [37] to extract 10 clips per video likewise. Since the pool5 representation of CaffeNet has the dimension of 256 × 6 × 6, we apply a max-pooling operation instead of the original global spatiotemporal pooling in three 3D CNNs to get a 512

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TMM.2020.3025661, IEEE Transactions on Multimedia

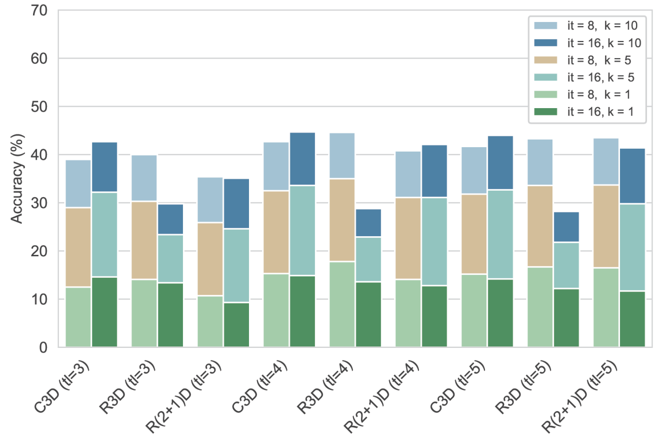


Fig. 6. Clip retrieval results of various models on UCF101, where it = 8, 16 and k = 1, 5, 10 respectively.

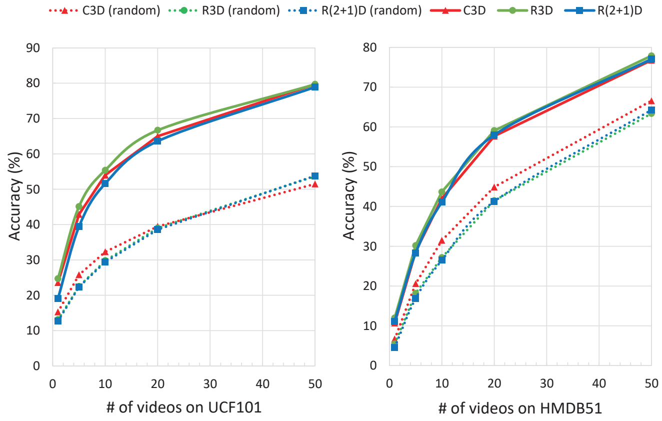


Fig. 7. Comparison of video retrieval results with various optimal models of 3D CNNs on UCF101 and HMDB51.

× 2 × 3 × 3 spatiotemporal representation, which is the same size as the other one. We conduct the validation process on split 1 of UCF101. The clips extracted from the test set are utilized to query clips from the training set. For each test clip, we compute the top k nearest neighbors in the training set by using cosine distance. It is considered as a correct prediction when the classes of top k nearest neighbors incorporate the class of test clip.

In Table II, we compare the experimental results for k = 1, 5, 10, 20, 50 with existing self-supervised methods on UCF101. The methods in top row adopt 2D CNNs, specifically, CaffeNet as the feature extractor, and 3D CNNs in the bottom are trained by the proposed self-supervised method. The results of random initialized 3D CNNs are also displayed for reference. It can be seen that 3D CNNs trained under the proposed self-supervised method outperform the randomly initialized counterparts and other self-supervised 2D CNNs particularly when k increases. Bückler et al. [37] shows competitive results when k is less than 10. They focus on an adaptive learning approach in which training samples and permutation sets can be adjusted by their model given the network states. It is promising to apply this approach here and get a promotion as well when assigning more clips per tuple. With the help of the task-based training method, we can see that the performance improves when tl becomes larger most of the time, which means that improve the difficulty of the task appropriately is helpful for learning meaningful representations. The contrast between R3D (16-8-3) and R3D (16-8-3)* indicates the valid improvement of feature representative capability of R3D trained by the sample-based curriculum learning strategy. In addition, we conduct the same experiment on split 1 of HMDB51, and the similar results are displayed in Table III. These feature extractors are purely trained on UCF101, that is, they are not theoretically exposed to videos from HMDB51, in spite of several videos that are duplicated in both datasets. Compared with the results of random initialization, the accuracy of the network trained by the proposed self-supervised method has been significantly improved.

To intuitively perceive the effect of clip interval length on various feature extractors, we also visualize the results of clip retrieval on UCF101 with it = 8, 16 and k = 1, 5, 10 in Fig 6. It is apparent that when it = 16, no matter how many clips are taken for each tuple, the retrieval accuracy of C3D is improved to some extent, while that of R3D is decreased, and there is no obvious change in R (2+1)D. Combined with the results from Table. I, we can get that although the proposed method can help to learn meaningful representations, the actual architectural design of 3D CNNs is also important under different task settings.

We further evaluate the learned representation on video level, the results are shown in Fig. 7. The video representation is the average of the 10 extracted clip features, and we can find consistent improvements for all kinds of 3D CNNs in both datasets. The top2 videos retrieved from UCF101 are visualized in Fig. 8. The leftmost columns are the videos used for the query, and the remaining columns show the top2 videos retrieved by various feature extractors. As we can see, the self-supervised trained 3D CNNs have the capability to retrieve videos with similar appearance or motion. For instance, when querying the video about playing violin, C3D network finds videos of playing cello and playing sitar, which are also musical performances and incorporating orchestral instruments. While for diving video, C3D and R3D networks also retrieve skijet videos, both of which are water-related sports.

In Fig. 9, we utilize the trained R3D (16-8-4) network to perform video retrieval experiment between UCF101 and HMDB51. We use a video from one dataset to retrieve videos from another dataset. Since the categories contained in two datasets are diverse, we cannot evaluate the performance of video retrieval quantitatively, but can only understand the state of video retrieval in a qualitative way. The sampled results show that the query video is more or less similar to the retrieved videos. For instance, the iconic objects or colors in the video are basically the same. When querying the video of biking, videos with bicycles, people and dark grey roads will appear. This further confirms the generality of the self-supervised trained model.

From the above experimental results, it can be seen that the task of clip order prediction indeed encourages 3D CNNs

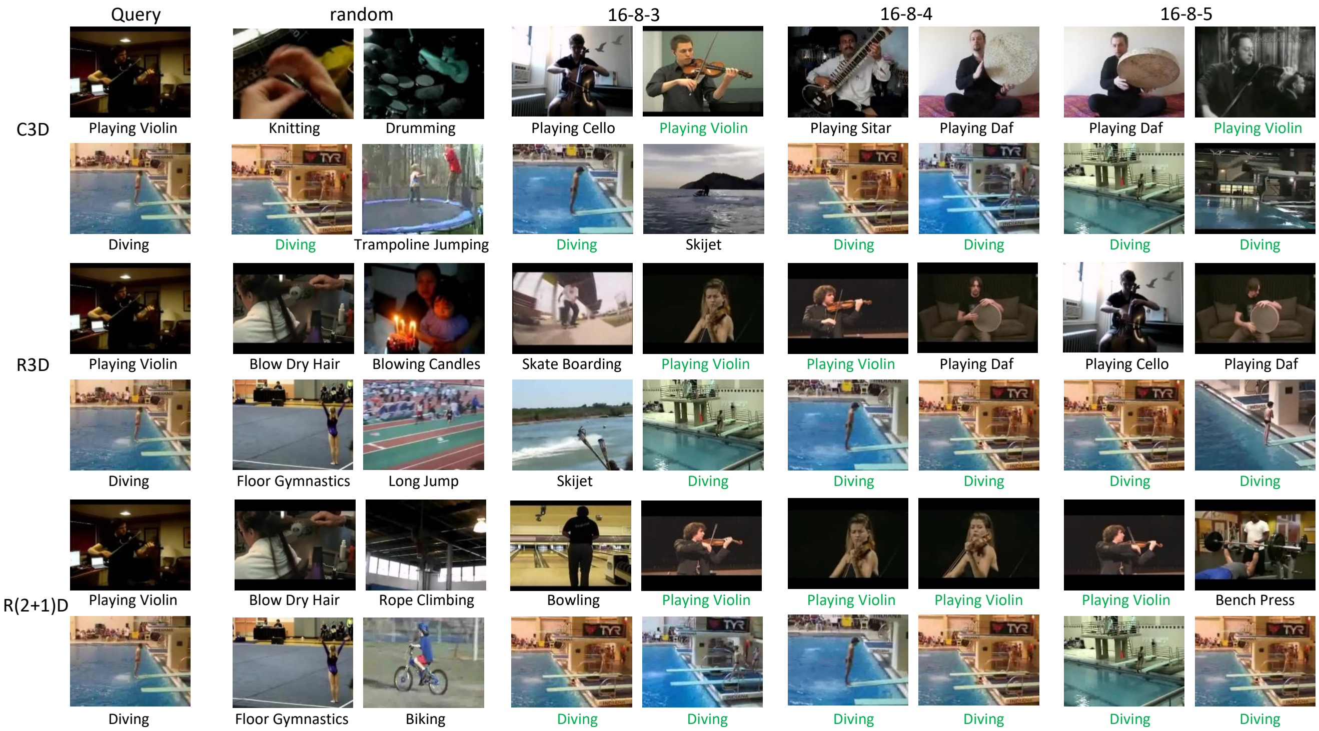


Fig. 8. Video retrieval samples on UCF101 with different models under various training settings. The first column contains the query videos from the test set, and the remaining columns represent the top2 videos retrieved from the training set. The actual class for each video is shown at the bottom, and the green color indicates that the prediction is correct.

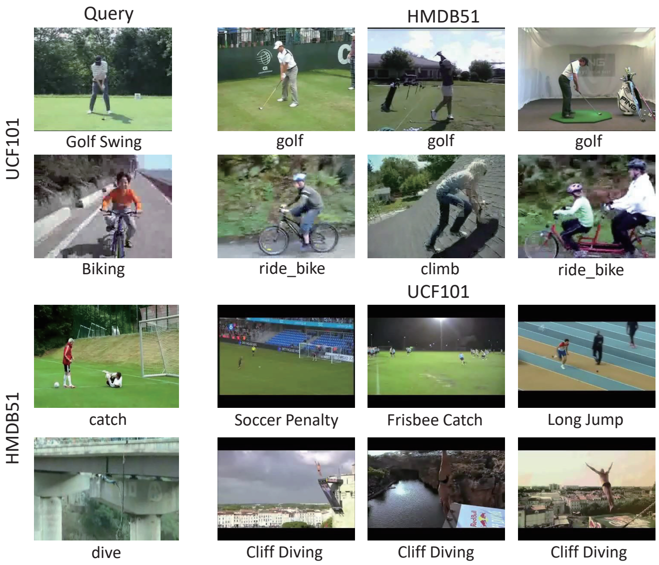


Fig. 9. The cross-dataset retrieval samples. The dataset names on the left and top represent the query and retrieval source, respectively. The actual class for each video is shown at the bottom.

TABLE IV
CLIP RETRIEVAL RESULTS ON UCF101 FROM DIFFERENT LAYERS, IN WHICH ALL 3D CNNS ARE TRAINED UNDER THE TASK SETTING OF cl = 16, it = 8 AND tl = 3.

Model	Layer	Top1	Top5	Top10	Top20	Top50
C3D	1	18.4	33.2	41.7	51.1	64.7
	2	20.5	35.2	43.3	52.6	65.3
	3	**23.3**	**40.5**	**49.6**	**59.6**	**72.7**
	4	18.6	36.5	46.0	56.5	70.9
	5	12.5	29.0	39.0	50.6	66.9
R3D	1	18.8	32.1	39.9	48.9	61.8
	2	19.9	34.9	43.4	52.6	65.4
	3	**20.6**	**37.3**	**46.4**	**56.0**	**69.6**
	4	16.9	34.6	44.2	55.0	**69.6**
	5	14.1	30.3	40.0	51.1	66.5
R(2+1)D	1	13.2	27.3	36.6	47.2	63.0
	2	**18.7**	**34.8**	**43.8**	53.4	66.6
	3	16.8	34.1	43.7	**54.0**	**67.8**
	4	12.8	29.2	39.3	50.7	67.0
	5	10.7	25.9	35.4	47.3	63.9

to learn more general spatiotemporal representations of video clips. Since the optimal performance is accomplished by applying curriculum learning strategies, we can conclude that the proposed strategies help to train 3D CNNs better and learn more meaningful spatiotemporal representations.

We also extract the features from each convolutional layer of C3D, R3D and R(2+1) networks, then pool and convert them into 9216-dimensional vectors as the clip features, and do nearest neighbor retrieval as before. The results are shown in Table IV. These 3D CNNs are all trained under the setting of cl = 16, it = 8 and tl = 3 on split 1 of UCF101. It can be seen that when k = 1, 5, 10, 20, 50, all highest accuracies obtained by three kinds of 3D CNNs networks appear in the layer 2 or 3. We speculate that the features in the middle layer are more suitable for the task of video retrieval since they contain both low-level and high-level features.

C. Action Recognition

Apart from using the trained model as the feature extractor directly, another possible application of these models is to do fine-tuning. 3D CNNs can be used for many downstream video-related tasks, here we take the most typical instance. In this part, we take the trained 3D CNNs as the initialized models and fine-tune them for action recognition on both UCF101 and HMDB51.

To obtain the action recognition results of the video, we adopt the settings of [33]. 10 clips are sampled from the video to get clip-level predictions, which are then averaged as the video-level prediction. To be specific, all three networks output a 512-dimension vector after the global spatiotemporal pooling layer, and we append a fully-connected layer with softmax on top of it as described in [33]. Only the fully-connected layers are randomly initialized, while the others are initialized from the self-supervised training one correspondingly. The hyperparameters and data preprocessing steps are the same as before. All networks are fine-tuned for 150 epochs.

Table V shows the comparison of the classification accuracy with other existing self-supervised methods. Since the experiments rely on massive computing resources, and the average classification accuracy over 3 splits is approximately the same as that of split 1, the experiments based on curriculum learning expanded in this paper are conducted on split 1 of UCF101 and HMDB51. For a more comprehensive comparison and reference, we not only show the results of the randomly initialized training of 2D CNNS and 3D CNNs, but also display the accuracy of the fine-tuned pre-training model from larger datasets. Comparing with 2D CNNs, the 3D CNNs trained from scratch can achieve higher accuracy than some 2D CNNs after fine-tuning, which proves the benefits of spatiotemporal modeling capability of 3D CNNs over videos. For the C3D network, the models trained by the proposed self-supervised learning method achieve a 5.3% and 8.7% improvement on UCF101 and HMDB51 compared with the randomly initialized model. For R3D and R(2+1)D networks, they gain lower accuracy if only trained from scratch on both datasets, but after the initialization of the proposed self-supervised training method, the accuracy of the two networks has been greatly improved, even exceeding that of C3D network. The R(2+1)D network get the state-of-the-art results among these self-supervised methods and achieves an improvement of 18.3% and 15.3% on UCF101 and HMDB51 respectively. Compared with R3D (16-8-3), R3D (16-8-3)* that trained under the sample-based curriculum learning strategy also obtains certain improvement on both datasets.

As mentioned before, the self-supervised training only utilizes split 1 of UCF101. Since all fine-tuned networks obtain the homologous improvement on both UCF101 and HMDB51, it demonstrates that the proposed self-supervised learning method has a wide range of applicability and favorable generalizability. Meanwhile, the application of both task-based and sample-based curriculum learning strategies improves the accuracy of action recognition, which proves that our strategies indeed reinforce the capability of 3D CNNs to learn more meaningful visual priors from the unlabeled videos.

TABLE V
ACTION RECOGNITION RESULTS ON UCF101 AND HMDB51. THE TOP ROWS ARE FRAME-BASED METHODS AND THE BOTTOM ROWS ARE CLIP-BASED METHODS. * MEANS THAT THE SAMPLE-BASED CURRICULUM LEARNING STRATEGY IS APPLIED.

Methods	UCF101	HMDB51
Shuffle&Learn [18]	50.2	18.1
VGAN [54]	52.1	-
Luo et al. [55]	53.0	-
OPN [20]	56.3	22.1
Jigsaw [15]	51.5	22.5
Büchler et al. [37]	58.6	25.0
ImageNet pre-trained	67.1	28.5
Wang et al. [38]	58.8	32.6
3D ST-puzzle [39]	63.4	30.8
Kinetics pre-trained	96.8	74.5
C3D (random)	61.6	23.2
C3D (VCP) [40]	68.5	32.5
C3D (16-8-3)	65.6	28.4
C3D (16-8-4)	66.9	31.8
C3D (16-8-5)	66.1	31.9
R3D (random)	54.4	21.5
R3D (16-8-3)	64.9	29.5
R3D (16-8-3)*	*65.5*	*31.4*
R3D (16-8-4)	66.0	28.0
R3D (16-8-5)	65.0	29.7
R(2+1)D (random)	56.2	22.0
R(2+1)D (PRP) [41]	72.1	35.0
R(2+1)D (16-8-3)	72.4	30.9
R(2+1)D (16-8-4)	72.2	**37.3**
R(2+1)D (16-8-5)	**74.5**	34.8

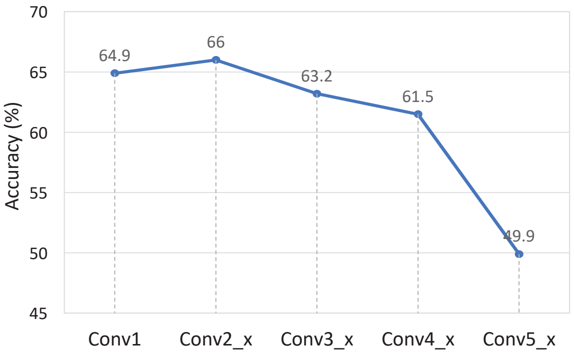


Fig. 10. The pre-trained R3D network parameters are used to initialize the model and the results are obtained by hierarchical fine-tuning on UCF101. Note that the convolutional layer in the figure represents the boundary between frozen and fine-tuned. The layers that are shallower than it are frozen, while both itself and the deeper layers are fine-tuned.

To explore the contribution of different layers when fine-tuning to action recognition, we carry out a hierarchical fine-tuning approach based on the R3D network. As shown in Fig 10, by selecting several layers as a split point, the shallower layers are frozen and deeper layers are fine-tuned with the same hyperparameters as before accordingly. We find that the fine-tuning of the network by freezing the Conv1 layer only outperforms other variants, including the fine-tuning of the entire R3D network. It is widely agreed that the generality and reusability of the extracted representation of a convolutional layer depend on the depth of the layer in the model, and the

TABLE VI

Experiment results on ActivityNet-QA. **“w/ pre-trained”** means the randomly initialized model pre-trained on action recognition task, **“w/ SS-VCOP”** means the model obtained by self-supervised learning on video clip order prediction task.

Models	Motion	Spat. Rel.	Temp. Rel.	Free						All
				Y/N	Color	Obj.	Loc.	Num.	Other	
C3D w/ PRE-TRAINED	2.0	8.5	2.1	57.0	30.6	17.9	14.2	46.2	28.2	29.0
C3D w/ SS-VCOP	3.7	9.2	2.1	60.0	32.0	19.8	17.1	45.0	28.9	**30.4**
R3D w/ PRE-TRAINED	2.8	9.6	3.1	57.9	29.3	16.4	10.9	46.7	28.2	29.2
R3D w/ SS-VCOP	2.5	9.4	1.5	59.0	31.9	18.9	16.3	45.9	30.0	**30.2**
R(2+1)D w/ PRE-TRAINED	1.8	8.7	2.9	59.6	29.6	14.2	11.9	47.4	26.3	29.1
R(2+1)D w/ SS-VCOP	2.4	10.4	2.4	59.8	31.0	19.2	14.2	46.4	29.2	**30.3**

shallower layer of the model always extracts more local and generic features. For this reason, it can be inferred that the R3D network trained under the clip order prediction task has already learned the most universal representations of video, such as visual edge, atomic motion and so on.

D. Video Question Answering

To further test the generalizability of the learned 3D CNNs, we evaluate them on the video question answering task, which requires the model to give correct answers in the context of videos and questions. The trained model can be used either as a feature extractor or a pre-training model in this task. Since fine-tuning requires lots of computing resources, here we just verify its effectiveness as a feature extractor. Considering the balance between model complexity and training time, we adopt the extended soft-attention (E-SA) model mentioned in [56], and conduct experiments on the ActivityNet-QA dataset [57].

The E-SA model first uses a Long Short-Term Memory (LSTM) network to encode the words in the question, then uses the encoded representation to attend on the extracted clip features, and finally uses the question and weighted video representation to predict the answer. The ActivityNet-QA dataset contains 58,000 QA pairs for 5,800 videos sampled from 20,000 videos in ActivityNet [58]. Each video corresponds to 10 pairs of questions and answers, which mainly include motion, spatial relation and temporal relation and free types. For free type, there are questions for yes/no, color, object, location, number and so on. The variety of questions and the complexity of the video make the dataset very challenging.

For video features, we sample 20 evenly distributed clips from each video and use the trained 3D CNNs to extract 20 feature vectors as the video representation. Note that each clip contains 16 frames and the dimension of the feature vector is 512. For the question feature, we transform each word through the embedding layer and input them to the LSTM networks to get the question representation. Following the settings in [56], we use GloVe [59] as the word embedding, and the hidden size of the LSTM networks is set to 300 in order to match that of the word embedding.

Two kinds of models are used as feature extractors in the experiment, including the randomly initialized model pre-trained with the action recognition task and the model pre-trained with video clip order prediction task under the 16-8-3 setting. In Table VI, we report the results under two types of models with C3D, R3D and R(2+1)D networks. Obviously, the optimal results are obtained from the feature extractor that is pre-trained directly with the task of video clip prediction, which proves that the features extracted by the proposed self-supervised method are more meaningful and transferable.

V. Conclusion

In this paper, we reveal the drawback of frame-based order prediction task and propose the clip order prediction task to learn the spatiotemporal representation of video better. Building on our previous work, we apply the curriculum learning in two ways, which considers both the sample and task difficulty, to enable the network to learn under more complex settings. We conduct experiments on three types of 3D CNNs and give comprehensive analyses about the clip order prediction task. We evaluate the capability of 3D CNNs as both fixed feature extractors and pre-trained models respectively in nearest neighbor retrieval, action recognition and video question answering tasks. Compared with the existing self-supervised learning methods, the models trained by the proposed method have achieved state-of-the-art results in nearest neighbor retrieval and action recognition tasks. Meanwhile, the promising results obtained in the video question answering application also prove that the models trained by the proposed method have good generalization. Besides, the improvements gained by C3D, R3D and R(2+1)D networks also show that the proposed methods are widely applicable for different 3D convolutional architectures.

While our study shows promising results in action recognition, it still cannot compete with those methods that fine-tuning from models trained with supervision on larger, labeled datasets such as Kinetics. The experimental results indicate that the clip order prediction task can not only assist 3D CNNs to learn better spatiotemporal representations of videos, but also provide good initialization parameters for other video-related tasks. Although the results are exciting, there are still open problems. For example, after the self-supervised training, the model performs better by using the shallow or middle layers than the whole network in different evaluation tasks, which means that deep layers are not trained well. Either the training methods or new proxy tasks are still needed in order to fully release the power of self-supervised learning. We expect that our work will stimulate more research interests in self-supervised learning with 3D CNNs.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[4] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, no. 2, pp. 303–338, 2010.

[5] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 4489–4497.

[6] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6299–6308.

[7] Z. Qiu, T. Yao, and T. Mei, "Learning spatio-temporal representation with pseudo-3d residual networks," in *proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 5533–5541.

[8] Y. Shi, Y. Tian, Y. Wang, and T. Huang, "Sequential deep trajectory descriptor for action recognition with three-stream cnn," *IEEE Transactions on Multimedia*, vol. 19, no. 7, pp. 1510–1520, 2017.

[9] J. Song, Y. Yang, Z. Huang, H. T. Shen, and J. Luo, "Effective multiple feature hashing for large-scale near-duplicate video retrieval," *IEEE Transactions on Multimedia*, vol. 15, no. 8, pp. 1997–2008, 2013.

[10] W. Zhang, S. Tang, Y. Cao, S. Pu, F. Wu, and Y. Zhuang, "Frame augmented alternating attention network for video question answering," *IEEE Transactions on Multimedia*, 2019.

[11] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Advances in neural information processing systems*, 2014, pp. 568–576.

[12] K. Hara, H. Kataoka, and Y. Satoh, "Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet?" in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 6546–6555.

[13] R. Goyal, S. E. Kahou, V. Michalski, J. Materzynska, S. Westphal, H. Kim, V. Haenel, I. Fruend, P. Yianilos, M. Mueller-Freitag *et al.*, "The" something something" video database for learning and evaluating visual common sense." in *ICCV*, vol. 1, no. 2, 2017, p. 3.

[14] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1422–1430.

[15] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *European Conference on Computer Vision*. Springer, 2016, pp. 69–84.

[16] G. Larsson, M. Maire, and G. Shakhnarovich, "Colorization as a proxy task for visual understanding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6874–6883.

[17] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2536–2544.

[18] I. Misra, C. L. Zitnick, and M. Hebert, "Shuffle and learn: unsupervised learning using temporal order verification," in *European Conference on Computer Vision*. Springer, 2016, pp. 527–544.

[19] B. Fernando, H. Bilen, E. Gavves, and S. Gould, "Self-supervised video representation learning with odd-one-out networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 3636–3645.

[20] H.-Y. Lee, J.-B. Huang, M. Singh, and M.-H. Yang, "Unsupervised representation learning by sorting sequences," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 667–676.

[21] D. Xu, J. Xiao, Z. Zhao, J. Shao, D. Xie, and Y. Zhuang, "Self-supervised spatiotemporal learning via video clip order prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10 334–10 343.

[22] P. Scovanner, S. Ali, and M. Shah, "A 3-dimensional sift descriptor and its application to action recognition," in *Proceedings of the 15th ACM international conference on Multimedia*. ACM, 2007, pp. 357–360.

[23] A. Klaser, M. Marszałek, and C. Schmid, "A spatio-temporal descriptor based on 3d-gradients," 2008.

[24] N. Dalal, B. Triggs, and C. Schmid, "Human detection using oriented histograms of flow and appearance," in *European conference on computer vision*. Springer, 2006, pp. 428–441.

[25] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition via sparse spatio-temporal features." VS-PETS Beijing, China, 2005.

[26] S. Sadanand and J. J. Corso, "Action bank: A high-level representation of activity in video," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 1234–1241.

[27] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *Proceedings of the IEEE international conference on computer vision*, 2013, pp. 3551–3558.

[28] C. Feichtenhofer, A. Pinz, and R. Wildes, "Spatiotemporal residual networks for video action recognition," in *Advances in neural information processing systems*, 2016, pp. 3468–3476.

[29] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2014, pp. 1725–1732.

[30] L. Wang, Y. Qiao, and X. Tang, "Action recognition with trajectory-pooled deep-convolutional descriptors," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 4305–4314.

[31] 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 *European conference on computer vision*. Springer, 2016, pp. 20–36.

[32] X. Wang, L. Gao, P. Wang, X. Sun, and X. Liu, "Two-stream 3-d convnet fusion for action recognition in videos with arbitrary size and length," *IEEE Transactions on Multimedia*, vol. 20, no. 3, pp. 634–644, 2017.

[33] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, "A closer look at spatiotemporal convolutions for action recognition," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 6450–6459.

[34] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.

[35] X. Wang, K. He, and A. Gupta, "Transitive invariance for self-supervised visual representation learning," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 1329–1338.

[36] B. Fernando, E. Gavves, J. Oramas, A. Ghodrati, and T. Tuytelaars, "Rank pooling for action recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 4, pp. 773–787, 2016.

[37] U. Buchler, B. Brattoli, and B. Ommer, "Improving spatiotemporal self-supervision by deep reinforcement learning," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 770–786.

[38] J. Wang, J. Jiao, L. Bao, S. He, Y. Liu, and W. Liu, "Self-supervised spatio-temporal representation learning for videos by predicting motion and appearance statistics," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4006–4015.

[39] D. Kim, D. Cho, and I. S. Kweon, "Self-supervised video representation learning with space-time cubic puzzles," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 8545–8552.

[40] D. Luo, C. Liu, Y. Zhou, D. Yang, C. Ma, Q. Ye, and W. Wang, "Video cloze procedure for self-supervised spatio-temporal learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[41] Y. Yao, C. Liu, D. Luo, Y. Zhou, and Q. Ye, "Video playback rate perception for self-supervised spatio-temporal representation learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6548–6557.

[42] S. Benaim, A. Ephrat, O. Lang, I. Mosseri, W. T. Freeman, M. Rubinstein, M. Irani, and T. Dekel, "Speednet: Learning the speediness in videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9922–9931.

[43] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th annual international conference on machine learning*. ACM, 2009, pp. 41–48.

[44] A. Pentina, V. Sharmanska, and C. H. Lampert, "Curriculum learning of multiple tasks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5492–5500.

[45] X. Zhang, P. Shapiro, G. Kumar, P. McNamee, M. Carpuat, and K. Duh, "Curriculum learning for domain adaptation in neural machine translation," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT*, 2019, pp. 1903–1915.

[46] L. Jiang, D. Meng, Q. Zhao, S. Shan, and A. G. Hauptmann, "Self-paced curriculum learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2015.

[47] M. P. Kumar, B. Packer, and D. Koller, "Self-paced learning for latent variable models," in *Advances in Neural Information Processing Systems*, 2010, pp. 1189–1197.

[48] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 1, pp. 221–231, 2012.

[49] 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.

[50] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, 2019, pp. 8024–8035.

[51] D. Masters and C. Luschi, "Revisiting small batch training for deep neural networks," *arXiv preprint arXiv:1804.07612*, 2018.

[52] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "Hmdb: a large video database for human motion recognition," in *2011 International Conference on Computer Vision*. IEEE, 2011, pp. 2556–2563.

[53] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014, pp. 675–678.

[54] C. Vondrick, H. Pirsiavash, and A. Torralba, "Generating videos with scene dynamics," in *Advances In Neural Information Processing Systems*, 2016, pp. 613–621.

[55] Z. Luo, B. Peng, D.-A. Huang, A. Alahi, and L. Fei-Fei, "Unsupervised learning of long-term motion dynamics for videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2203–2212.

[56] D. Xu, Z. Zhao, J. Xiao, F. Wu, H. Zhang, X. He, and Y. Zhuang, "Video question answering via gradually refined attention over appearance and motion," in *Proceedings of the 25th ACM international conference on Multimedia*. ACM, 2017, pp. 1645–1653.

[57] Z. Yu, D. Xu, J. Yu, T. Yu, Z. Zhao, Y. Zhuang, and D. Tao, "Activitynet-qa: A dataset for understanding complex web videos via question answering," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 9127–9134.

[58] F. Caba Heilbron, V. Escorcia, B. Ghanem, and J. Carlos Niebles, "Activitynet: A large-scale video benchmark for human activity understanding," in *Proceedings of the ieee conference on computer vision and pattern recognition*, 2015, pp. 961–970.

[59] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

Jun Xiao received the Ph.D. degree in computer science and technology from the College of Computer Science, Zhejiang University, Hangzhou, China, in 2007. He is currently a professor with the College of Computer Science, Zhejiang University. His current research interests include computer animation, multimedia retrieval, and machine learning.

Lin Li received the B.S. degree from the College of Software, Qufu Normal University, Qufu, China, in 2019. Currently, she is an M.Eng. student in the College of Software Technology, Zhejiang University, Hangzhou, China. Her current research interests include deep learning and computer vision.

Dejing Xu received the B.S. degree from the College of Computer Science, Zhejiang University in 2015. He is currently a Ph.D. student in the College of Computer Science at Zhejiang University. His research interests include machine learning and multimedia analysis.

Chengjiang Long is currently a Computer Vision Researcher/Senior R&D Engineer in Computer Vision Team at Kitware Inc. since Feb 2016. He works as an Adjunct Professor at University at Albany, SUNY since Aug 2018 and also worked as an Adjunct Professor at Rensselaer Polytechnic Institute (RPI) from Jan 2018 to May 2018. He received the M.S. degree in Computer Science from Wuhan University in 2011 and a B.S degree in Computer Science and Technology from Wuhan University in 2009.He got his Ph.D. degree in Computer Science from Stevens Institute of Technology in 2015. Prior to joining Kitware, he worked at NEC Labs America and GE Global Research as a research intern in 2013 and 2015,respectively.To date, he has published over 40 papers including top journals such as T-PAMI and IJCV, top international conferences such as CVPR,ICCV and AAAI, and owns 1 patent. He is also the reviewer for more than 20 top international journals and conferences. His research interests involve various areas of computer vision, machine learning, artificial intelligence and computer graphics. He is a member of IEEE and AAAI.

Jian Shao received the Ph.D. degree in signal and information processing from Institute of Acoustics, Chinese Academy of Sciences, Beijing, China. He is currently an Associate Professor with the College of Computer Science and Technology, Zhejiang University. His research interests include unstructured data management, cross media computing, and cognitive decision service.

Shifeng Zhang Ph.D., senior engineer of Hangzhou Hikvision Digital Technology Co., Ltd., expert in the field of artificial intelligence of China electronic science and Technology group, expert of the 14th five-year plan, enterprise tutor of Hangzhou Dianzi University. His research interests include analog integrated circuits, artificial intelligence chips, and computer vision. etc. He has published more than 11 papers including TCSII, EDL, EL and other international conferences, and owns 15 Chinese invention patents and 1 American invention patent. He also wrote and published a practical course on analog integrated circuit and digital integrated circuit design tools for general higher education during the 13th five-year plan (2016-2020), which won the first prize of Zhejiang province science and technology progress award in 2018.

Shiliang Pu is the chief expert of Hikvision, and the director of Hikvision Research Institute, enjoying the special allowance of the state council. He has received the Qiu Shi outstanding youth achievement transformation award from the China Association for Science and Technology (CAST), young and middle-aged experts with outstanding contribution expert of Zhejiang province, the first prize of Zhejiang province science and technology award (first complete adult) and so on. His main research interests include artificial intelligence and large visual data. Several visual perceptive technologies developed by Dr. Pu have been widely used in public security, finance, transportation, justice, retailing, smart city, etc. In recent years, Dr. Pu leads the R&D team making constant breakthrough in image and video recognition, and have won several the first prizes in many international authoritative AI competitions, such as KITTI Benchmark, MOT Challenge, ImageNet Large Scale Visual Recognition Challenge, and ICDAR Robust Reading Competition. Moreover, Dr. Pu has published over 10 papers in international conferences and journals, and obtained 17 authorized invention patents.

Yueting Zhuang received his B.Sc., M.Sc. and Ph.D. degrees in Computer Science from Zhejiang University, China, in 1986, 1989 and 1998 respectively. From February 1997 to August 1998, Yueting Zhuang was a visiting scholar at Prof. Thomas Huang's group, University of Illinois at Urbana-Champaign. Currently, He is a full professor of the College of Computer Science, Zhejiang University. His research interests mainly include artificial intelligence, multimedia retrieval, computer animation and digital library.