Self-supervised Spatiotemporal Learning via Video Clip Order Prediction

Dejing Xu[1] Jun Xiao[1] Zhou Zhao[1] Jian Shao[1] Di Xie[2] Yueting Zhuang[1]
[1]Zhejiang University [2]Hikvision Research Institute
{xudejing,junx,zhaozhou,jshao,yzhuang}@zju.edu.cn; xiedi@hikvision.com

Abstract

We propose a self-supervised spatiotemporal learning technique which leverages the chronological order of videos. Our method can learn the spatiotemporal representation of the video by predicting the order of shuffled clips from the video. The category of the video is not required, which gives our technique the potential to take advantage of infinite unannotated videos. There exist related works which use frames, while compared to frames, clips are more consistent with the video dynamics. Clips can help to reduce the uncertainty of orders and are more appropriate to learn a video representation. The 3D convolutional neural networks are utilized to extract features for clips, and these features are processed to predict the actual order. The learned representations are evaluated via nearest neighbor retrieval experiments. We also use the learned networks as the pre-trained models and finetune them on the action recognition task. Three types of 3D convolutional neural networks are tested in experiments, and we gain large improvements compared to existing self-supervised methods.

1. Introduction

3D convolutional neural networks (CNNs) have been explored for the action recognition task in many previous works [38, 2, 30]. While many high-level tasks have been proposed such as captioning [3] and question answering [43], action recognition is always significant for its foundation status. However, compared to the progress made by 2D CNNs over images, the improvements of 3D CNNs over videos are much slower. Until recently, the 2D CNNs that take both the RGB and flow streams [33] as inputs still behave competitively with 3D CNNs in action recognition. The primary reason is that most of the existing video datasets are relatively small-scale which may not be able to optimize the immense number of parameters in 3D CNNs [13]. ImageNet [5] plays an important role in various tasks in the image domain, but there is a lack of similar dataset in the video domain. In [13], several successful model architectures in image classification are extended and trained on Kinetics [2] video dataset. The authors conclude that 3D CNNs and Kinetics may have the potential to contribute to significant progress in fields related to various video tasks as the 2D CNNs and ImageNet do.

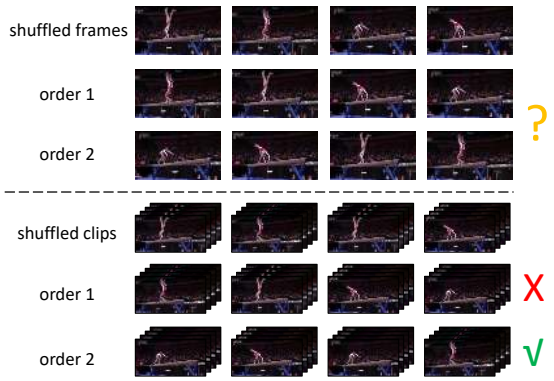

Figure 1. **Illustration of the necessity to use clips**. The top three rows show examples of frame-based order prediction. For the shuffled frames, both order 1 and order 2 has the same correctness since we cannot tell the direction of the gymnast in the balance beam. However, for the clip-based order prediction in the bottom rows, the judgment is simpler. The dynamics in each clip will help to reveal the correct order.

Though nowadays the large-scale video datasets begin to appear [12, 2], annotating new video datasets are always required to tackle the problem in new domains. It needs a wealth of resources and meticulous design to annotate such one. Therefore it is valuable if we can leverage the unlabeled videos to facilitate learning. Self-supervised learning is one kind of technique where the supervisory signal can be obtained easily from the data itself. Researches have been done in this area, in which both the images and videos are exploited. For image data, there exist self-supervised tasks such as predicting relative positions of image patches [6], solve jigsaw puzzles [27], image inpainting [29] and image color channel prediction [21]. Since the particular prop-

erty of the video is temporal information, recent works also attempt to leverage the temporal relations among frames, such as order verification [26, 9] and order prediction [22] of frames.

The existing self-supervised works that utilize the video have the framework as follows: first use 2D CNNs to extract features from the frames, then concatenate these features and predict the verification result or the actual order of the input frames. The whole framework is trained end-to-end. After the training, the learned 2D CNNs can be used as an image feature extractor or finetuned to other tasks such as image classification and detection. Compared to order verification [26, 9], order prediction [22] contains much richer supervisory signals and shows better performance in several validation experiments.

However, the order is not uniquely determined when referring to frames merely. As Figure 1 shows, given the shuffled frames, it is hard to tell the correct order of frames since both directions of the gymnast in the balance beam seems possible. To mitigate the defect of this, [22] groups both the forward and backward orders as the same class. It is a compromise under the circumstances that only frames and 2D CNNs are used. In contrast, we propose to directly use clips and 3D CNNs to make the task more well-defined. From the figure, we can get that if the shuffled clips are provided, the order will be more specific because of the inner dynamics contained in each clip. Besides, the 3D CNNs are always believed to be hard to optimize due to the lack of labeled videos [13]. With the assistance of the clip order prediction task, the 3D CNNs can leverage numerous videos without any labels. The 3D CNNs thus can be easily pre-trained to adapt to different video distributions in new application domains, which is a prerequisite to gain good performance.

In this paper, we integrate 3D CNNs into the clip order prediction task. First several fixed-length clips are sampled from the video and shuffled randomly, then 3D CNNs are used to extract features for these clips, and finally, a simple neural network is employed to predict the actual order of the shuffled clips. The learned 3D CNNs can be either used as a clip feature extractor or a pre-trained model to be finetuned on other tasks. The main contributions of the paper can be summarized as follows:

- We propose to utilize 3D CNNs and video clips to do order prediction task, which is more consistent with the video dynamics;
- C3D, R3D, and R(2+1)D networks are tested to prove that the proposed self-supervised learning framework is available widely;
- We validate the learned representations via the nearest neighbor retrieval experiments, and also finetune the learned 3D CNNs on action recognition, both experiments show promising results;

The rest of the paper is organized as follows. We first review related works in Section 2, then the details of the proposed method are explained in Section 3. In Section 4, the implementation and results of the experiments are provided and analyzed. Finally, we conclude our works in Section 5.

2. Related Work

In this section, we first introduce the recent progress in action recognition, then we discuss recent works on self-supervised representation learning.

Action Recognition Action recognition is one of the classic problems in computer vision area. The basic pipeline to tackle the task is first extracting features then doing classification. From traditional hand-crafted features [32, 18, 4, 7, 31, 41] to deep neural networks derived features [44, 11, 8, 35, 15, 23], the improvements are obvious.

Lots of researches relating to applying 2D CNNs on videos are proposed since the breakthrough on image classification made by AlexNet [19]. Many of them extract features for frames sampled from the video and fuse these features as a representation of the video. In [33], the input video is decomposed into the spatial stream and optical flow stream. Each stream are processed by a deep 2D CNNs and the prediction is made by late fusion. [17] proposes three kinds of fusion methods to integrate the temporal information of the video. It also implements multiresolution by splitting the inputs frames into context stream and fovea stream. Both streams are processed using 2D CNNs.

3D CNNs is a natural extension of 2D CNNs over temporal data such as videos. [38] proposes the C3D architecture where 3D convolution kernels are stacked followed by fully connected layers. In [2], the successful 2D CNNs trained on ImageNet are converted into 3D CNNs via inflating all the filters and pooling kernels. The proposed I3D model is based on Inception-v1 [37] and take both RGB and flow as inputs. The ResNet [14] architecture is also extended in [30, 39, 13] by adapting the 2D convolution kernels to 3D ones. [30] designs three types of bottleneck building blocks and interleaves these blocks to form the P3D ResNet. The decomposition of 3D convolution to 2D spatial and 1D temporal convolutions are adopted in both [30, 39]. In [13], they focus on the training of very deep 3D CNNs from scratch and indicate that deeper 3D CNNs trained on large datasets can be more effective.

Self-Supervised Representation Learning With the availability of large-scale data and abundant computing power, deep neural networks show promising results in computer vision tasks such as image classification and video recognition. CNNs are one of the critical factors to gain such improvements. It learns hierarchical representations of the input data which can be used in other related tasks directly or after finetuning. Though there exists a large

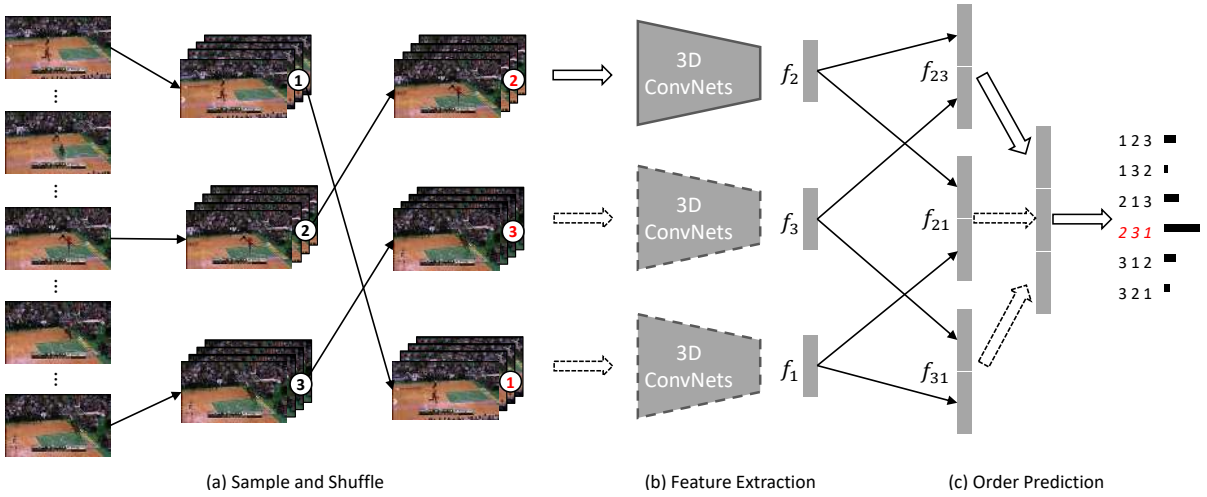

Figure 2. **Overview of Clip Order Prediction Framework**. (a) Sample and Shuffle: Sample non-overlapping clips and shuffle them to a random order. (b) Feature Extraction: Use the 3D ConvNets to extract the feature of all clips. The 3D ConvNets is not pre-trained in any datasets. (c) Order Prediction: The extracted features are pairwise concatenated, and fully connected layers are placed on top to predict the actual order. The dashed lines mean that the corresponding weights are shared among clips. The framework can be trained end-to-end, and the 3D ConvNets can be used as a video feature extractor or pre-trained weights after training.

amount of data, it takes great efforts to annotate such massive data, which is necessary to enable the supervised training of the CNNs. Self-supervised representation learning is one type of techniques that learn representations by solving a surrogate task, where the supervisory signals can be obtained for free.

There exist many works that leverage the unlabeled images. [6] proposes to learn image representation by predicting the relative positions between two image patches. The patches are sampled from the same image in eight spatial arrangements. In [27], nine tiles are extracted from the image and shuffled according to a predefined permutation set to make jigsaw puzzles. The permutation set is determined via a greedy algorithm based on Hamming distance between all possible permutations. [29] use an encoder-decoder architecture to tackle the image inpainting task, while [21] focus on image colorization, where the color components of an image are predicted given its intensity.

Videos are also utilized because of the temporal coherence and dynamics they have. [42] exploits different self-supervised approaches to learn representations invariant to inter-instance and intra-instance variations among object patches. The object patches are extracted from unlabeled videos using motion cues. [10] use the ranking machines to capture the evolution of appearances among frames, and the learned functional parameters can be used as the video representation. In [26, 9], the chronological order of frames in the video are exploited, and the method is required to tell whether the frame sequence is ordered or not. [22] propose another related task, in which the actual order of input frame sequences should be predicted. The task is formulated as a multi-class classification problem, and both the forward and backward orders are grouped into the same class. Since the number of possible permutations or orders are exploded when patches or frames are increased, the permutations are always predefined as we mentioned before [27]. While in [1], a reinforcement learning algorithm is used to propose permutations for the 2D CNNs training.

Though the above works make use of videos for self-supervised representation learning, the actual inputs are frames. As a result, the learned CNNs are only capable of extract features for still images. We extend the order prediction task proposed in [22] from frames to clips, which can help to leverage the strength of 3D CNNs and inner dynamics of clips. The details of our method are explained in the next section.

3. Clip Order Prediction

In this section, we will first give a brief overview of the proposed method, then each part of the method will be clarified in detail. Figure 2 presents the overall framework, which is composed of mainly three procedures. In sample and shuffle, several clips are uniformly sampled and shuffled; in feature extraction, 3D CNNs are used to extract features for the clips, and all 3D CNNs used here shared same weights; in order prediction, we resolve the task via classification as in [22]. The extracted features are pairwise concatenated and forwarded through two linear layers, after which a softmax layer is applied to output the probability distribution over the possible orders.

In order to make the following descriptions more clear,

we first introduce several definitions. A clip is made up of continuous frames sampled from the video with the size $c\times l\times h\times w$, where c is the number of frame channels, l is the number of frames, h and w indicates the height and width of frames. A 3D convolution kernel has the size $t \times d \times d$, where t is the temporal depth and both d are spatial size. We define a tuple of ordered clips as $\boldsymbol{C} = \langle c_1, c_2, \ldots, c_n\rangle$, and the features extracted by 3D CNNs are represented as $\boldsymbol{F} = \langle \boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_n\rangle$. The subscript here indicates the chronological order.

3.1. Sample and Shuffle

For N clips, there exist $N!$ possible orders. The number of orders overgrows with the increase in the number of clips. For example $7! = 5040$, which already makes the classification task very hard. Previous works [27, 1] select several particular orders from all possible orders either determinately or adaptively. Since clip order prediction is just a proxy task and we focus on the learning of 3D CNNs, the task should be solvable. Otherwise, if the whole task is too hard to solve, almost nothing can be learned. We restrict the number of clips between 2 to 5, which makes the number of order classes less than 120.

The clips are sampled uniformly from the video, with an interval of m frames. The clips are forced to be non-overlapping, which can avoid the situation that the whole framework tackles the task by comparing lower characteristics such as texture and color. For an extreme case, if the clips are overlapped by 1 frame, a simple comparison of frames pixels can solve the task.

After the clips are sampled, they are shuffled to form the input data while the actual order is served as the target, as shown in Figure 2 (a). The shuffle step should be random, and no particular permutations are preferred. During the training step, the number of generated samples belonging to each order class is roughly the same.

3.2. Feature Extraction

Once the shuffled clips are prepared, 3D CNNs are used to extract features for each clip. The same 3D CNNs are used for all clips in one tuple, as Figure 2 (b) shows. We choose three different 3D CNNs as the feature extractor, which are C3D [38], R3D and R(2+1)D [39] networks. The structure of distinct convolution blocks are presented in Figure 3. We will discuss the architecture of each network concretely in the following.

C3D [38] The model is a natural extension from 2D CNNs over videos. 3D CNNs is well-suited for spatiotemporal learning since it can model the temporal information and dynamics of the video [15, 38]. C3D network includes 8 convolution layers stacked one by one, with 5 pooling layers interleaved, and followed by two fully connected layers terminally. The size of all convolution kernels are $3\times3\times3$, which is the best practice gained from their experiments.

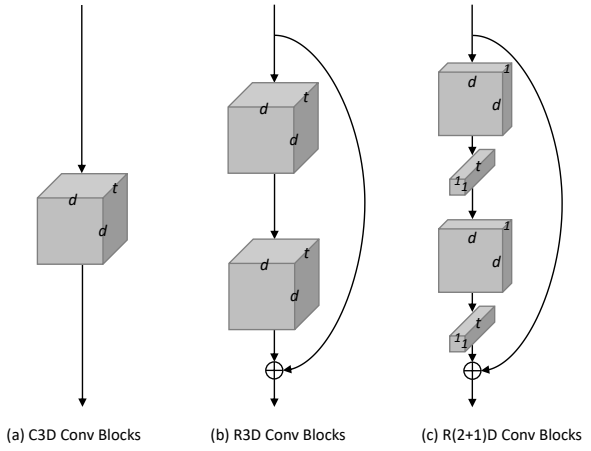

Figure 3. **Three types of 3D Conv Blocks**. (a) C3D Conv Blocks: the classic 3D convolution kernel with size $t \times d \times d$, which are stacked to form the C3D network. (a) R3D Conv Blocks: classic 3D convolution kernels with a shortcut connection. (c) R(2+1)D Conv Blocks: the 3D kernel are decomposed 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.

R3D [39] Residual learning principle [14] is a milestone for the architecture design of 2D CNNs. ResNet pushes the performance of many image-related tasks such as classification, detection, and segmentation to state-of-the-art. R3D network is the 3D CNNs with residual connections. The operations of the basic convolution 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 $\boldsymbol{x}_o$ stands for the input and output of the block, $\mathcal{F}$ is the 3D convolution operation, and $\mathcal{H}$ is a fucntion to scale the $\boldsymbol{x}_i$ to the size of $\boldsymbol{x}_o$ when necessary. The convolution block consists of two 3D kernels, with batch normalization and ReLU layers appended. There are 5 convolution layers in total, the specification can be refered in Table 1 of [39].

R(2+1)D [39] The 3D convolution kernel is convolved over a volume both spatially and temporally. The procedure can be refactored by first applying spatial convolution then temporal convolution [30, 36]. The concrete operations in 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, $\mathcal{S}$ stands for spatial convolution, $\mathcal{T}$ stands for temoporal convolution, and $\mathcal{H}$ is the same function as mentioned before. The overall architecture is the same as R3D, except that more ReLU layers are inserted in the block, which means the number of nonlinearties are

doubled while the number of parameters are almost the same. The R(2+1)D network also leads to state-of-art results on four action recognition benchmarks.

Both R3D and R(2+1)D networks use a global spatiotemporal pooling layer to aggregate the activations after the convolution layers. The obtained vector is treated as the extracted feature for the input clip. To have a fair comparison, we modify the original C3D implementation to following the same design. The batch normalization is also added after each convolution layer.

3.3. Order Prediction

The order prediction is formulated as a classification task. The input is a tuple of clip features, and the output is a probability distribution over different orders. We use a simple multi-layer perceptron, and the extracted features are pairwise concatenated, which is proved to be better for both order prediction and the learning of underlying feature extractors [22]. 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.

Assume a tuple contains 3 clips, after shuffling, we get $\boldsymbol{C} = \langle c_2, c_3, c_1 \rangle$ and the corresponding extracted features $\boldsymbol{F} = \langle \boldsymbol{f}_2, \boldsymbol{f}_3, \boldsymbol{f}_1 \rangle$. As Figure 2 (c) shows, the features are first pairwise concatenated as $\langle \boldsymbol{f}_{23}, \boldsymbol{f}_{21}, \boldsymbol{f}_{31} \rangle$ and then transformed to form a tuple of three vectors which capture the relationship between each clip. These vectors are concatenated again and a fully-connected layer with softmax are applied over to output the final 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 correctness of the prediction is measured using cross-entropy as follows,

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

where y_i and p_i are the probability of the sample belonging to order class i in groundtruth and prediction, and C is number of all possible orders. The loss $\mathcal{L}$ is then back-propagated to optimize the whole framework. When the framework is trained to predict the order of clips, the 3D CNNs are trained to extract the meaningful features of clips meanwhile.

4. Experiments

In this section, we will first describe the concrete settings of the clip order prediction experiments and their results, then the learned 3D CNNs are evaluated both quantitatively and qualitatively via nearest neighbor retrieval in Section 4.1 and action recognition in Section 4.2.

Though our self-supervised method is designed to learn from unlabeled videos, we choose to experiment based on UCF101 [34] since its diverse enough and well organized. Besides, it has many reported results to compare. HMDB51 [20] are also utilized to test the generalizability of the proposed method. The details of both datasets are described in the following.

UCF101 is an action recognition dataset of realistic action videos, collected from YouTube, having 101 action categories. The action categories can be divided into five types: 1) Human-Object Interaction 2) Body-Motion Only 3) Human-Human Interaction 4) Playing Musical Instruments 5) Sports. 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.

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 actions categories can be grouped in five types: 1) General facial actions 2) Facial actions with object manipulation 3) General body movements 4) Body movements with object interaction 5) Body movements for human interaction. The dataset contains 6,849 clips divided into 51 action categories, each containing a minimum of 101 clips.

We use PyTorch [28] to implement the whole framework. Since the C3D network contains 8 convolution layers, we implement the R3D network with no repetitions in conv{2-5}_x, which results in 9 convolution layers in total. C3D network is also modified by replacing the two fully connected layers with a global spatiotemporal pooling layer as used in the R3D network. R(2+1)D network follows the same architecture as the R3D network, with only 3D kernels decomposed. Dropout layers are applied between fully-connected layers with $p = 0.5$. All nonlinearities are ReLU.

The split 1 of UCF101 is used to train and test our self-supervised learning method. We choose clip length as 16 frames since most 3D CNNs [30, 39, 13, 38] requires a 16-frames clip as input. The interval is set to be 8 frames, which is required to avoid trivial solutions of the task. For tuple length, 3 clips per tuple are reasonable since 2 clips order prediction is more like an order verification, while more than 3 tuples become a relatively hard task. This decision is also based on the conclusion from [27] that *a good self-supervised task is neither simple nor ambiguous*. On-the-

3D CNNs	C3D	R3D	R(2+1)D
Accuracy	68.5	68.4	64.2

Table 1. **Clip order prediction results on UCF101**. C3D, R3D and R(2+1)D networks are trained with clip order prediction framework separately.

fly data augmentation is used 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, then randomly cropped to 112×112 during training. During validation or testing, the clip is cropped in the center.

To optimize the framework, we use mini-batch stochastic gradient descent. Memory consumption is always a trouble when training neural networks with large batches, especially for 3D CNNs. Recently [25] shows that small mini-batch sizes provide more up-to-date gradient calculations and yields more stable and reliable training. Thus we choose 8 tuples per batch. The learning rate is set to 0.001, while the momentum is 0.9 and weight decay is 0.0005. The training process lasts for 300 epochs, and the model with the lowest validation loss is saved to be the best model.

As Table 1 shows, with the clip as 16 frames, the interval as 8 frames and the tuple as 3 clips, the clip order prediction task can reach an accuracy higher than 64% for all three 3D CNNs. Considering that the accuracy of random guessing for the task is 16.7%, the framework indeed learns to analyze the content of clips and reason the order out. We also test the 4 clips per tuple with the C3D network. During the training phase, the accuracy increases very slowly but continuously, while a larger learning rate causes unstable training at present. As a result, for the following validation experiments, we use the ones trained under 3 clips per tuple setting if not particularly specified.

4.1. Nearest Neighbor Retrieval

As mentioned before, to accomplish the clip order prediction task, the framework needs to analyze and understand the content of clips. As the feature extractor, the 3D CNNs are trained together with the whole framework. To evaluate the learned representations, we choose the nearest neighbor retrieval method since it is also used in [1, 26].

We basically follow the experiment settings in [1]. The split 1 of UCF101 is used for validation. In their experiment, 10 frames are extracted per video, and the pool5 layer of CaffeNet [16] is selected as the representation. In our experiment, we extract 10 clips per video likewise. Since the pool5 representation of CaffeNet has the dimension of $256 \times 6 \times 6$, we apply a max pooling operation instead of the original global spatiotemporal pooling in three 3D CNNs to get a $512 \times 2 \times 3 \times 3$ spatiotemporal representation, which is the same size as the other one. The clips extracted from the test set are used to query the clips from the training set. The cosine distance of representations between the query clip and all clips in the training set are computed. When the class of a test clip appears in the classes of k nearest training clips, it is considered to be correctly predicted.

We show the accuracies for $k = 1, 5, 10, 20, 50$ and compare with the other self-supervised methods on UCF101 in Table 2. The top row in the table are those which use 2D CNNs, specifically, CaffeNet as the feature extractor, and the bottom shows 3D CNNs trained by our self-supervised method. The results of random initialized 3D CNNs are also presented. As we can see, our self-supervised trained 3D CNNs perform better than random initialized ones and self-supervised trained 2D CNNs especially when k becomes larger. Büchler et al. [1] presents a competitive result when k is less than 10. Their method focuses on adjusting the permutation set based on network states, which we can expect to apply in our method and get a promotion as well when we use more clips per tuple. We also test the trained 3D CNNs on split 1 of HMDB51, the results are presented

Methods	Top1	Top5	Top10	Top20	Top50
Jigsaw [27]	19.7	28.5	33.5	40.0	49.4
OPN [22]	19.9	28.7	34.0	40.6	51.6
Büchler et al. [1]	*25.7*	*36.2*	*42.2*	*49.2*	*59.5*
C3D (random)	16.0	22.5	26.7	31.4	39.3
C3D	12.5	29.0	39.0	50.6	**66.9**
R3D (random)	10.5	17.2	21.5	27.0	36.7
R3D	14.1	**30.3**	**40.0**	**51.1**	66.5
R(2+1)D (random)	10.2	17.3	21.9	27.7	38.5
R(2+1)D	10.7	25.9	35.4	47.3	63.9

Table 2. **Frame and clip retrieval results on UCF101.** The methods in top row are based on 2D CNNs while 3D CNNs in our framework are presented in bottom row.

Methods	Top1	Top5	Top10	Top20	Top50
C3D (random)	7.7	12.5	17.3	24.1	37.8
C3D	7.4	22.6	**34.4**	48.5	**70.1**
R3D (random)	5.5	11.3	16.5	23.8	37.2
R3D	7.6	**22.9**	**34.4**	**48.8**	68.9
R(2+1)D (random)	4.6	11.1	16.3	23.9	39.3
R(2+1)D	5.7	19.5	30.7	45.8	67.0

Table 3. **Clip retrieval results on HMDB51.** The 3D CNNs used here are self-supervised trained on split 1 of UCF101 merely.

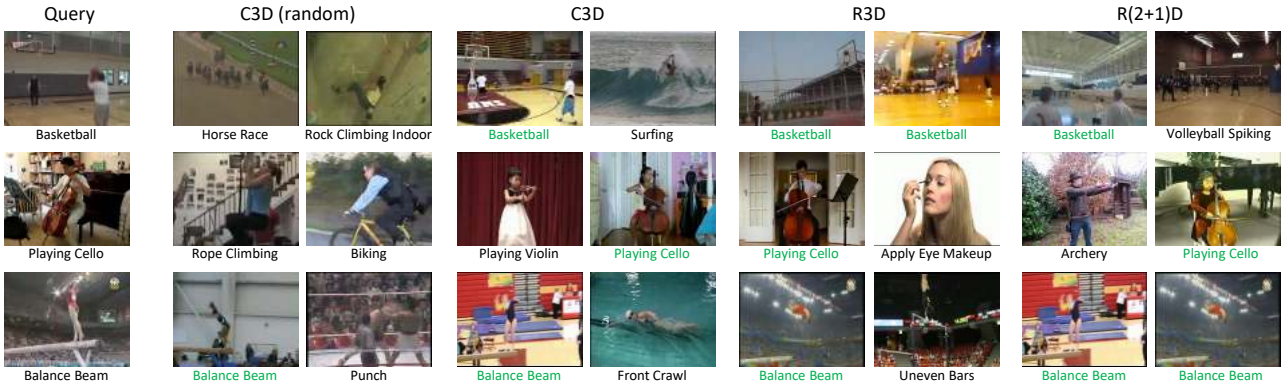

Figure 4. **Video retrieval samples on UCF101.** The first column contains query clips from the test split, and the remaining columns indicate top2 nearest clips retrieved by different trained models from the training split. The class of each video is displayed in bottom.

in Table 3. Since these feature extractors are trained on UCF101 merely, which means they never see any videos from HMDB51 theoretically (actually there may be some duplicated videos between two datasets since we find one in samples). The results are even better which indicates the generalizability of our trained feature extractors.

Note that for $k = 1$, the self-supervised trained 3D CNNs do not show noticeable improvement. We find that the top1 accuracy is sensitive to the clip sample rate. To further evaluate the learned representation, we adopt the same experiment in video level. The video representation is the average of the 10 extracted clip features. As shown in Figure 5, compared to randomly initialized networks, the accuracy of all trained 3D CNNs are higher for all k consistently in both datasets, among which R3D and C3D networks are slightly better than R(2+1)D network.

To have an intuitive understanding of the video retrieval results, we also visualize the top2 retrieved videos from UCF101 in Figure 4. The videos are represented by their central frames, and the actual classes are displayed under the figures. The leftmost columns are videos used for the query, and the remaining columns show top2 retrieved videos by different feature extractors. As we can see, the self-supervised trained 3D CNNs can find videos with similar meanings. For query video of basketball, R(2+1)D network also finds volleyball spiking video which both is also sports and contains balls. For balance beam video, R3D network also retrieves uneven bars video, which is also gymnastics.

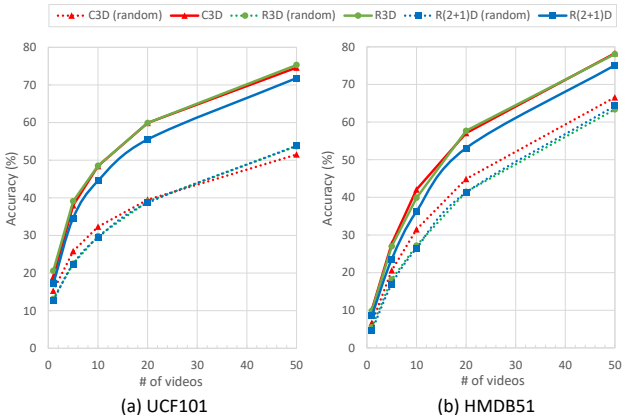

Figure 5. **Video retrieval results.** The clip features are averaged to form the video representation.

To test the generalizability of our trained feature extractors, we also adopt cross-dataset video retrieval between UCF101 and HMDB51 using the trained R3D network. Since the two datasets have different classes, we cannot evaluate the video retrieval performance quantitatively. Several results are showed in Figure 6 to understand the status qualitatively. We use one dataset for query and the other dataset for retrieval. Though the classes are not the same, we can see that the retrieved videos have the classes relating to the query one more or less.

From the above experimental results, we can conclude that the clip order prediction task indeed encourages the 3D CNNs to learn more general spatiotemporal representations for clips and videos.

4.2. Action Recognition

In addition to acting as feature extractors, we can also take the trained 3D CNNs as initializations and finetune the networks on other supervised tasks. Here we finetune the network on action recognition task on both UCF101 and HMDB51.

The three networks all 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 in [39]. Only the fully-connected layers are randomly initialized, other layers are initialized from the self-supervised training one correspondingly. The hyperparameters and data pre-

Figure 6. **Cross-dataset retrieval samples.** The dataset names in left and top indicates the query and retrieval sources correspondingly. The class of each video is displayed in bottom.

processing steps are the same as before. All networks are finetuned for another 150 epochs. To get the action recognition result for a video, we follow the method from [39]. 10 clips are sampled from the video to get clip predictions, which are then averaged to obtain the video prediction.

We report the average classification accuracy over 3 splits and compare with other finetuning results from existing self-supervised methods in Table 4. The training from randomly initialized 2D CNNs and 3D CNNs are reported for reference. We also show the accuracies from finetuned models which are pre-trained on large supervised datasets such as ImageNet and Kinetics. As we can see, the 3D CNNs trained from scratch already beats several 2D CNNs after finetuning, which indicates the capability of 3D CNNs over videos. The finetuned C3D network gives 4.0% and 5.2% improvements on UCF101 and HMDB51 compared to the randomly initialized one. R3D and R(2+1)D networks gain lower accuracy if only trained from scratch on both datasets. However, with the initialization from our self-supervised training, the R3D and R(2+1)D networks achieve even better accuracies than the C3D network. The best-performed R(2+1)D network gets 16.2% and 8.9% promotions on both datasets than random initialization. The model also beats the state-of-the-art from Büchler et al. [1] by 13.8% on UCF101 and 5.9% on HMDB51.

Since our initialization networks are only trained on split 1 of UCF101, the same improvements gained by all finetuned networks on 3 splits of the UCF101 and HMDB51 datasets prove that our self-supervised learning technique is widely applicable and has good generalizability.

Method	UCF101	HMDB51
Shuffle&Learn [26]	50.2	18.1
VGAN [40]	52.1	-
Luo et al. [24]	53.0	-
OPN [22]	56.3	22.1
Jigsaw [27]	51.5	22.5
Büchler et al. [1]	58.6	25.0
ImageNet pre-trained	*67.1*	*28.5*
C3D (random)	61.6	23.2
C3D	65.6	28.4
R3D (random)	54.4	21.5
R3D	64.9	29.5
R(2+1)D (random)	56.2	22.0
R(2+1)D	**72.4**	**30.9**
Kinetics pre-trained	*96.8*	*74.5*

Table 4. **Action recognition results on UCF101 and HMDB51.** The top row is frame-based methods and the bottom row is clip-based methods.

5. Conlusions

In this paper, we introduce the clip order prediction task to leverage the inner dynamics of the video better. The task is very suitable for 3D CNNs which can model the spatiotemporal information. We experiment with three types of 3D CNNs and evaluate them using nearest neighbor retrieval and finetuning on action recognition. From the experimental results, we can get that the clip order prediction task can encourage the 3D CNNs to learn a general spatiotemporal representation as well as a good initialization. We hope that our work will inspire more research interests on self-supervised learning of 3D CNNs. While our study shows promising results, the finetuning from supervised pre-training on larger datasets such as Kinetics still act as the best. Future direction will be the combination of our method with more unlabeled videos and search for diverse task settings.

6. Acknowledgements

National Key Research and Development Program of China (2017YFB0203001), Zhejiang Natural Science Foundation (LR19F020002, LZ17F020001), National Natural Science Foundation of China (61572431), Chinese Knowledge Center for Engineering Sciences and Technology, Key R&D Program of Zhejiang Province (2018C03055), the Fundamental Research Funds for the Central Universities and Joint Research Program of ZJU & Hikvision Research Institute.

References

[1] Uta Buchler, Biagio Brattoli, and Bjorn Ommer. Improving spatiotemporal self-supervision by deep reinforcement learning. In *Proceedings of the European Conference on Computer Vision*, pages 770–786, 2018. 3, 4, 6, 8

[2] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017. 1, 2

[3] Long Chen, Hanwang Zhang, Jun Xiao, Liqiang Nie, Jian Shao, Wei Liu, and Tat-Seng Chua. Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5659–5667, 2017. 1

[4] Navneet Dalal, Bill Triggs, and Cordelia Schmid. Human detection using oriented histograms of flow and appearance. In *Proceedings of the European conference on computer vision*, pages 428–441, 2006. 2

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 1

[6] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1422–1430, 2015. 1, 3

[7] Piotr Dollár, Vincent Rabaud, Garrison Cottrell, and Serge Belongie. Behavior recognition via sparse spatio-temporal features. In *IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, pages 65–72, 2005. 2

[8] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2625–2634, 2015. 2

[9] Basura Fernando, Hakan Bilen, Efstratios Gavves, and Stephen Gould. Self-supervised video representation learning with odd-one-out networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3636–3645, 2017. 2, 3

[10] Basura Fernando, Efstratios Gavves, José Oramas, Amir Ghodrati, and Tinne Tuytelaars. Rank pooling for action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(4):773–787, 2017. 3

[11] Rohit Girdhar, Deva Ramanan, Abhinav Gupta, Josef Sivic, and Bryan Russell. Actionvlad: Learning spatio-temporal aggregation for action classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 971–980, 2017. 2

[12] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The something something video database for learning and evaluating visual common sense. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2, page 8, 2017. 1

[13] Kensho Hara, Hirokatsu Kataoka, and Yutaka 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*, pages 6546–6555, 2018. 1, 2, 5

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2, 4

[15] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):221–231, 2013. 2, 4

[16] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678, 2014. 6

[17] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014. 2

[18] Alexander Klaser, Marcin Marszałek, and Cordelia Schmid. A spatio-temporal descriptor based on 3d-gradients. In *19th British Machine Vision Conference*, 2008. 2

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 2

[20] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. Hmdb: a large video database for human motion recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2556–2563, 2011. 5

[21] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich. Colorization as a proxy task for visual understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6874–6883, 2017. 1, 3

[22] Hsin-Ying Lee, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. Unsupervised representation learning by sorting sequences. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 667–676, 2017. 2, 3, 5, 6, 8

[23] An-An Liu, Yu-Ting Su, Wei-Zhi Nie, and Mohan Kankanhalli. Hierarchical clustering multi-task learning for joint human action grouping and recognition. *IEEE transactions on pattern analysis and machine intelligence*, 39(1):102–114, 2017. 2

[24] Zelun Luo, Boya Peng, De-An Huang, Alexandre Alahi, and Li Fei-Fei. Unsupervised learning of long-term motion dynamics for videos. In *Proceedings of the IEEE Conference*

on Computer Vision and Pattern Recognition, pages 2203–2212, 2017. 8

[25] Dominic Masters and Carlo Luschi. Revisiting small batch training for deep neural networks. *arXiv preprint arXiv:1804.07612*, 2018. 6

[26] Ishan Misra, C Lawrence Zitnick, and Martial Hebert. Shuffle and learn: unsupervised learning using temporal order verification. In *Proceedings of the European Conference on Computer Vision*, pages 527–544, 2016. 2, 3, 6, 8

[27] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *Proceedings of the European Conference on Computer Vision*, pages 69–84, 2016. 1, 3, 4, 5, 6, 8

[28] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 5

[29] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2536–2544, 2016. 1, 3

[30] Zhaofan Qiu, Ting Yao, and Tao Mei. Learning spatio-temporal representation with pseudo-3d residual networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5533–5541, 2017. 1, 2, 4, 5

[31] Sreemanananth Sadanand and Jason J Corso. Action bank: A high-level representation of activity in video. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1234–1241, 2012. 2

[32] Paul Scovanner, Saad Ali, and Mubarak Shah. A 3-dimensional sift descriptor and its application to action recognition. In *Proceedings of the 15th ACM international conference on Multimedia*, pages 357–360, 2007. 2

[33] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014. 1, 2

[34] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012. 5

[35] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using lstms. In *International conference on machine learning*, pages 843–852, 2015. 2

[36] Lin Sun, Kui Jia, Dit-Yan Yeung, and Bertram E Shi. Human action recognition using factorized spatio-temporal convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4597–4605, 2015. 4

[37] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 2

[38] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015. 1, 2, 4, 5

[39] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6450–6459, 2018. 2, 4, 5, 7, 8

[40] Carl Vondrick, Hamed Pirsiavash, and Antonio Torralba. Generating videos with scene dynamics. In *Advances In Neural Information Processing Systems*, pages 613–621, 2016. 8

[41] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *Proceedings of the IEEE international conference on computer vision*, pages 3551–3558, 2013. 2

[42] Xiaolong Wang, Kaiming He, and Abhinav Gupta. Transitive invariance for self-supervised visual representation learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1329–1338, 2017. 3

[43] Dejing Xu, Zhou Zhao, Jun Xiao, Fei Wu, Hanwang Zhang, Xiangnan He, and Yueting Zhuang. Video question answering via gradually refined attention over appearance and motion. In *Proceedings of the 25th ACM international conference on Multimedia*, 2017. 1

[44] Zhongwen Xu, Yi Yang, and Alex G Hauptmann. A discriminative cnn video representation for event detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1798–1807, 2015. 2