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: DOI10.1109/TCSVT.2021.3114209, IEEE Transactions on Circuits and Systems for Video Technology

Self-supervised Representation Learning for Videos by Segmenting via Sampling Rate Order Prediction

Jing Huang, Yan Huang, Qicong Wang, Wenming Yang, *Senior Member, IEEE* and Hongying Meng, *Senior Member, IEEE*

***Abstract*—Self-supervised representation learning for videos has been very attractive recently because these methods exploit the information inherently obtained from the video itself instead of annotated labels that is quite time-consuming. However, existing methods ignore the importance of global observation while performing spatio-temporal transformation perception, which highly limits the expression capabilities of the video representation. This paper proposes a novel pretext task that combines the temporal information perception of the video with the motion amplitude perception of moving objects to learn the spatio-temporal representation of the video. Specifically, given a video clip containing several video segments, each video segment is sampled by different sampling rates and the order of video segments is disrupted. Then, the network is used to regress the sampling rate of each video segment and classify the order of input video segments. In the pre-training stage, the network can learn rich spatio-temporal semantic information where content-related contrastive learning is introduced to make the learned video representation more discriminative. To alleviate the appearance dependency caused by contrastive learning, we design a novel and robust vector similarity measurement approach, which can take feature alignment into consideration. Moreover, a view synthesis framework is proposed to further improve the performance of contrastive learning by automatically generating reasonable transformed views. We conduct benchmark experiments with several 3D backbone networks on two datasets. The results show that our proposed method outperforms the existing state-of-the-art methods across the three backbones on two downstream tasks of human action recognition and video retrieval.**

***Index Terms*—Self-supervised, video representation learning, pretext task, contrastive learning.**

I. Introduction

VIDEO representation has always been an important basis for completing many video understanding tasks such as video retrieval, action recognition, etc. In order to obtain a more discriminative video representation, many powerful network architectures based on spatio-temporal feature extraction [1], [2], [3], [4], [5] are designed. In addition, the great success of ImageNet [6] inspires people to create large-scale video datasets, such as Kinetics [7] and ActivityNet [8]. Despite the considerable progress made, video representation learning under a supervised manner still has two limitations. (1) The production of video datasets is a labor-intensive task. Therefore, insufficient training data hinders the application of many video understanding tasks. (2) The video representation learned from labels has poor generalization and does not have the potential to be well applied to multiple downstream tasks.

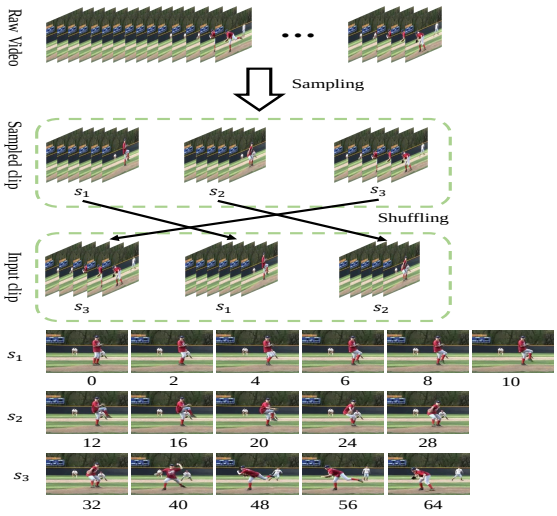


Fig. 1. Illustration of the proposed segment sample rate order prediction task. As shown in the figure, each sampled video clip is composed of n video segments. Here, we adopt the strategy of 3 video segments, each video segment is composed of a random number of video frames (we limit the shortest video segment length candidate to be 3), and each video segment is sampled by the different sample rates. Then, disrupt the order of video segments. The subscripts of the images in the bottom three rows represent the index of the image in the raw video.

Recently, many self-supervised representation learning methods for video are proposed [9], [10], [11], [12], [13], [14], [15]. The convolutional neural network first performs pretexts on massive unlabeled videos in a self-supervised manner. The supervision signals of pretext tasks come from the video data itself and do not require any manual annotation. Then, there are two main ways to apply the pre-trained network to downstream tasks. One is to treat the pre-training as the initialization of network parameters while the other is to directly use the pre-trained network as a feature extractor for downstream tasks.

Early self-supervised methods design pretext tasks based

This work was supported by the Shenzhen Science and Technology Projects under Grant No. JCYJ20200109143035495 and JCYJ20180306173210774. *(Corresponding author: Qicong Wang.)*

J. Huang, Y. Huang and Q. Wang are with the Department of Computer Science and Technology, Xiamen University, Xiamen 361005, China, and also with the Shenzhen Research Institute, Xiamen University, Shenzhen, 518000 China (e-mail: qcwang@xmu.edu.cn).

W. Yang is with the Shenzhen International Graduate School/Department of Electronic Engineering, Tsinghua University, Shenzhen, 518055, China (e-mail: yang.wenming@sz.tsinghua.edu.cn).

H. Meng is with the Department of Electronic and Electrical Engineering, Brunel University London, London, UK, UB83PH (e-mail: hongying.meng@brunel.ac.uk).

on the spatial transformation of images, including solving a jigsaw puzzle [16], image repairing, and predicting the rotation angles of images [17]. However, due to the lack of consideration of the temporal relationship between video frames, the learned video representation cannot be effectively applied to video understanding tasks. To introduce advanced temporal information for video representation, many methods based on video order verification/prediction [9], [18], [19],[20] are proposed. Specifically, given a video after temporal transformation, these methods leverage the network to perform a binary classification task to determine whether the order of input video is correct or predict the order of input video clips. However, such methods are limited by the single auxiliary task structure, which results in much-unmined information in the video. As for temporal information, many works reveal that the motion amplitude of moving objects is also a kind of meaningful supervision signal [11], [12]. These supervision signals can drive the network to pay attention to moving objects whose movement trends are most obvious, so it can achieve impressive performance. However, solving these tasks cannot help the network to pay attention to the global features, and it may cause the learned representation to be too local. Although the above-mentioned methods can achieve encouraging results, they are still restricted by a single-task framework and local observations. Other methods include using the corresponding relationship of multiple data streams to guide the network to generate a video representation with good spatio-temporal discrimination [21], [22], [23], and learning spatio-temporal representations through statistical regression of motion and appearance [24]. The methods using multiple data streams can introduce additional data calculation and storage consumption. Contrastive learning has become a hot topic for self-supervised representation learning [12], [25]. However, these proposed methods do not explore which view transformation makes contrastive learning more effective.

In this paper, we assume that a good self-supervised video representation learning method should be simple and intuitive and the learned video representation can globally reflect the entire content of the input video clip. Under this assumption, we propose a novel self-supervised video representation learning method. This method classifies the order of unlabeled input video segments and predicts the sampling rate of each video segment. The network can learn the spatio-temporal representation of the video by executing this method. Besides, we introduce contrastive learning to enhance the discrimination of the video representation. Given a video, we sample several video segments according to different sampling rates on the real timeline. As shown in Fig. 1, each video segment may consist of a different number of video frames, and the order of video segments is disrupted. Then, the neural network is used to predict the order of video segments and the sampling rate of each video segment. We sample two video clips from the same video as content-related positive pairs, and the video clips sampled from other videos are treated as negative pairs. We propose a view synthesis framework based on information decoupling, which minimizes the mutual information between the input and synthesized samples while preserving the motion information of the input samples as much as possible. Then, we maximize the mutual information between the encoded feature of positive pairs by minimizing the info-NCE loss [26]. To better preserve the encoded temporal information, we also propose a novel feature similarity measurement method, referred to as shuffle similarity (SS).

The contributions of this work can be summarized as follows.

- We propose a simple and effective self-supervised representation learning method for video to predict the order and the sampling rate of input video segments. By analyzing the video in segments, this method can effectively extract high-level temporal semantic information, capture the moving objects, and avoid the extracted spatio-temporal features from being too local.
- Contrastive learning is introduced to enhance the discrimination of video representations and we further propose a novel feature vector similarity measurement method to eliminate the appearance dependence caused by contrastive learning. In addition, a view synthesis framework is proposed to further the representation learning.
- Three network architectures are used to evaluate the proposed method in two downstream tasks. The experimental results show that our approach achieves state-of-the-art performance on two datasets, reflecting the superiority of our proposed method.

II. Related work

A. Learning from Video Content

Self-supervised video representation learning methods that learning from video content leverage the semantic information in the video to design several transformations as supervision signals which can drive the network to extract rich semantic information. The first motivation is to allow the network to learn the semantic information in the video during the process of data reconstructing. For example, the network is used to generate unsampled video frames based on the context [27], colorize the grayscale video [28], and predict future frames based on the given video frames [29], [30]. However, most of the above-mentioned self-supervised video representation learning methods based on data reconstruction are pixel-level dense predictions, which require a large number of network parameters to train the auxiliary task itself. It is not conducive to obtaining a robust video representation since too many network parameters are used to solve specific auxiliary tasks.

Because pixel-level dense prediction task is too heavy for the network, many works focus on designing simpler and more effective pretext tasks. Luo et al. [10] leverage the network to detect the type of spatio-temporal transformation based on the contextual semantic information to obtain the semantic information of the raw video. Xu et al. [31] train a deep model to estimate the geometric deformation applied to the original sketches. Compared to it, our method is more suitable for video tasks. Guo et al. [32] use the 2D heat map as the intermediate supervised signal for 3D hand pose estimation. In comparison with this, our method fuses three self-supervised signals and makes avoiding collapse solution into consideration. Feichtenhofe et al. [33] propose a two-stream structure

to fuse the features of different time resolutions for video representation learning, which is applied to action recognition tasks with great success. Inspired by this, many self-supervised video representation methods based on temporal resolution prediction [11], [12], [34], [13] are proposed. The essence of this kind of method is to use the motion amplitude of the moving objects in the video as the supervision signal to drive the network to pay more attention to the moving objects, so impressive results can be achieved.

B. Learning from Video Order

Self-supervised video representation learning methods that learning from video order are dedicated to making full use of the temporal information. Temporal information is the key to improving the performance of video understanding tasks. Therefore, many self-supervised video representation learning methods are proposed to utilize the temporal information of the video as much as possible. One way of utilizing the temporal information is to leverage the network to verify whether the input frame sequence with the correct order [18], [19], [20]. However, these methods often treat the temporal order verification as a binary classification task, where the network does not need much temporal information to complete the verification task. In order to increase the state space for solving the pretext tasks, the second way on utilizing the temporal information is to recognize the order of the input frame sequence [9], [35], [36]. In comparison, the second way needs more temporal information to complete the task, so it can achieve better performance.

Although self-supervised video representation learning methods based on temporal correlation can extract video temporal information to a certain extent, they may still encounter ambiguity problems sometimes. For example, for the stirring action, both clockwise stirring and counterclockwise stirring are possible. It is difficult for the self-supervised video representation learning framework which only relies on a single auxiliary task to learn feature representations that sufficiently reflect action changes.

C. Learning from the Correspondence between Multiple Data Streams

Self-supervised video representation learning methods utilize the correspondence between multiple data streams so that the generated video representation can put the correlation of various modalities of data into consideration. Nicolas et al. [37] automatically collect training set from Web videos according to the given textual description and establish the mapping between the textual description and video representation while our method does not require a lot of textual description. Mahendran et al. [22] design an auxiliary task based on the correlation verification of RGB video frames and optical flow. Sayed et al. [21] use a dual-stream structure to extract features from RGB video frames and optical flow data respectively and verify whether these two input data streams are related to each other. Gan et al. [38] treat the prediction of optical flow between two consecutive frames as an auxiliary task. However, the background jitter greatly affects recognition performance because optical flow estimation is a dense prediction task. To this end, Wang et al. [24] design a self-supervised video representation learning framework based on spatio-temporal statistics by using optical flow estimation algorithms from coarse to fine. In order to further utilize the correspondence of multimodal data, Tao et al. [23] utilize contrastive learning to train the network, taking different modalities of the same video as positive pairs and samples from different videos as negative pairs. Han et al. [25] believe that RGB data and optical flow data can complement in the feature space of each other, and propose a joint training framework to learn video representation.

The self-supervised video representation learning methods based on multi-modal data correspondence cleverly leverage the correspondence between different modalities as the supervision signal. Since the data of different modalities have complementary information, this kind of method based on the perception of the corresponding relationship of multi-modal data can reduce appearance dependence to a certain extent, and generate a more robust video representation.

III. Proposed Method

The proposed self-supervised video representation learning method is described in detail in this section. Firstly, we provide an overview of the proposed method. Secondly, we give some necessary formulations. Thirdly, each component of the proposed method is explained one by one. Finally, we outline the entire learning framework and explain the logical connections between the components.

TABLE I
Notations and Definitions

Notations	Definitions
v_i	a video in dataset
$x_{v_i}^k$	a video clip sampled from v_i
T	the number of frames included in the video clip
f_θ	the parameters of backbone
r_{v_i}	the representation of video v_i
h	the logits of order prediction
p_i	the probability of x_{v_i} is the i_{th} permutation
y_i	the ground truth of order prediction
$\pounds_{order}$	the error of order prediction
α	the regression of sampling rate
$\hat{\alpha}$	the ground truth of sampling rate
$\pounds_{sr}$	the error of sampling rate regression
$h(\cdot)$	an MLP projection head
$\psi_{x_{v_i}^k}$	the feature coding of $x_{v_i}^k$
$Sim(\psi_1, \psi_2)$	the similarity between ψ_1 and ψ_2
$\pounds_{ctr}$	the loss of contrastive
D	the distance matrix
dis	the minimum matching distance of two vectors

A. Overview

The overview of our proposed method is shown in Fig. 2. Firstly, we sample a video clip $query$ from the raw video, and feed it into the view synthesis module to generate a new view as a candidate of key. Next, we resample a video clip used as another candidate and randomly select a sample from the candidates as the key. Then, the $query$ and key are fed into the encoder Q and the encoder K respectively for feature

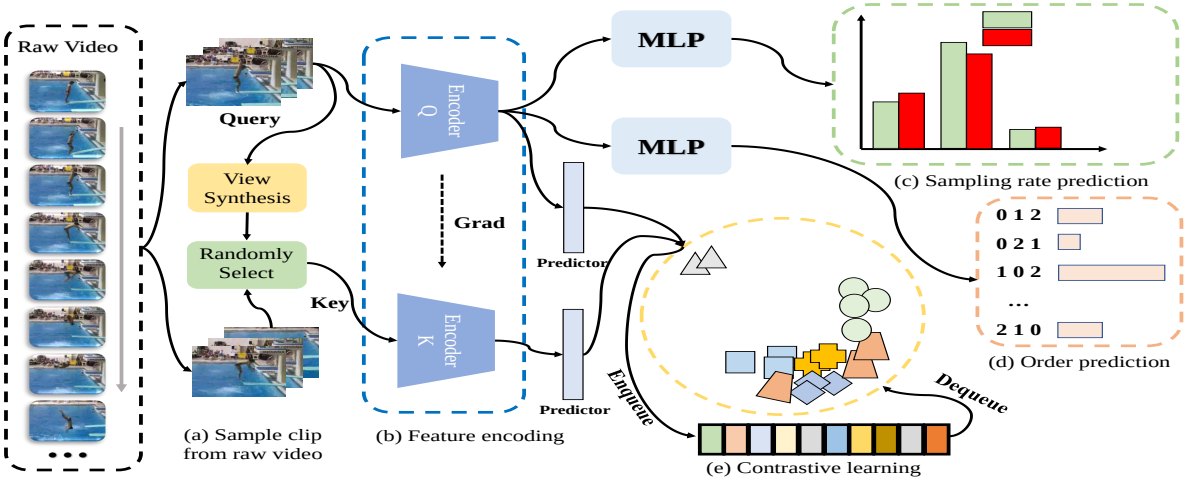


Fig. 2. Overview of our proposed method. (a) Sampling two video clips by the different sampling rates and orders from the raw video. One is used as $query$, and another is used as a candidate for the key. After that, $query$ is fed into view synthesis to generate a transformed view which is used as another candidate for the key. Then randomly select a sample from the candidates as the key. (b) We use two encoders to encode $query$ and key into feature representations containing rich temporal and spatial semantic information respectively. The encoder K does not perform backpropagation, it updates the gradient through momentum update and gradually approaches the encoder Q. (c) According to the feature representation of the encoder Q, the sample rate of video segments in the $query$ are regressed. (d) According to the feature representation encoded by the encoder Q, the order of video segments in the $query$ is classified. (e) $Query$ and key are respectively encoded into 512-dimensional feature vectors by encoder Q and encoder K. A fully connected layer projects them into the feature space for contrastive learning with the negative samples stored in the dynamic queue Q.

encoding. Note that the encoder K does not perform back-propagation during the training process, the parameters are updated by momentum gradient. The feature vector encoded from the $query$ is firstly used for sampling rate prediction and order prediction, and then used for contrastive learning together with the feature vector encoded from the key.

B. Formulation

Given a video v_i, we sample a video clip $x_{v_i} \in R^{C\times T\times H\times W}$ from this video, where C is the number of channels of the frame, T is the number of frames included in the video clip x_{v_i}, H and W indicate the height and width of the video frames, respectively. As shown in Fig. 1, x_{v_i} consists of n video segments s, that is $x_{v_i} = s_{v_i}^{(1)}, s_{v_i}^{(2)}, \ldots, s_{v_i}^{(n)}$. For each video segment $s_{v_i}^{(k)} = v_i^1, v_i^2, \ldots$, where v_i^j indicates the j_{th} frame of video v_i. We make l_i the length of $s_{v_i}^k$, then $T = \sum_{i=1}^{N} l_i$. After encoded by spatio-temporal convolutional neural network f_θ, the video clip can be represented as $f_\theta(x)$. Refer to Table I for the definition of the notations appearing in this paper.

C. Video Segment Order Prediction

Temporal information can reflect the changes of video frames in the timeline and is also an important source of information for video understanding tasks. Some works use 2D CNN [39] to extract the features of video frames separately and concatenate them to form a video representation. Then the video representation is utilized to perform an order verification task. However, these methods have an obvious drawback. Considering only a single frame of video results in the learned video representation, it cannot effectively reflect the spatiotemporal dynamics of the video over a period of time, which can limit the performance on downstream tasks. To this end, we extend the order verification task based on a single frame to the video segment order recognition. Essentially, our method takes advantage of the continuity of the video on the timeline. Therefore, uniform sampling may cause the network only focus on the connection between two video segments and only a trivial solution can be obtained in this way. To increase the state space of the auxiliary task solution, we sample each video segment with a random sampling rate. Besides, to prevent the network from focusing on the connections between video segments too much, we sample each video segment with a random length. In this way, the length of video segments is not equal, and the network can not solve the video segment order prediction problem by only focusing on the continuity of fixed-position. We do not encode each segment separately and then concatenate these segment features to predict the order of the segments like[9]. Because it can cause the network to focus on video segment-level features, instead of the video clip-level features. In addition, we made the temporal dynamics more significant by increasing the sampling interval.

We model the video segment prediction as a classification task. For a video clip with N segments, there are $N!$ kinds of possible permutations, each permutation corresponds to a category of the classification task. In our method, video segment order prediction is only an auxiliary task and should not be too difficult. Otherwise, many network parameters are used to solve the auxiliary task, which is not conducive to learning robust video representation. Therefore, we limit the number of video segments between 2 and 5. For the video segment order prediction, the input of the model is a video clip containing N segments, and the output of the model is the probability distribution of different orders. Here, we use multiple layer perception (MLP) to establish the mapping relationship between the video feature representation and the probability distribution of different orders. The specific process

is represented as follows:

$$r_{v_i} = f_\theta(x_{v_i}) \tag{1}$$

$$h = W_2(W_1 r_{v_i} + b_1) + b_2 \tag{2}$$

$$p_i = \frac{e^{h_i}}{\sum_{j=1}^{C} e^{h_j}} \tag{3}$$

where r is the video representation generated by the network backbone, W and b represent the parameters of the linear classifier, h is the logits of order prediction, C is the number of all permutations, and p_i is the probability that the order of the input video clip is the i_{th} permutation. We specify y as the ground truth of the order of input video segments, and use cross-entropy loss to measure the error of order prediction, then:

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

D. Video Segment Sampling Rate Prediction

Video is a kind of three-dimensional data, and a large amount of high-level semantic information is hidden in the spatio-temporal dynamics of the video. We hope that the network can pay more attention to the moving objects in the video during the representation learning stage because the movement trends of the moving objects often best reflect the content of the video, while reducing the negative effects of noisy background to some extent. Generally, a moving object is composed of pixels with the most significant movement trend in the video. This observation inspires us to design an auxiliary task that utilizes the motion amplitude of the moving objects as a supervision signal.

In a video play at a constant speed, the motion amplitude of moving objects per unit time is approximately equal. Based on this observation, we approximate the different motion amplitudes of the moving objects by different sampling rates of video. The larger the sampling rate is, the greater the motion amplitude of the moving objects. On the contrary, the smaller the motion amplitude. Yao et al. [11] propose a self-supervised video representation learning method based on playback rate perception. They model playback rate perception as a classification problem. Unlike them, we feel that it is more intuitive to model the motion amplitude prediction of moving objects as a regression problem instead of a classification problem. The semantic information required to solve the regression problem is significantly more than that of the classification problem. Besides, the method of Yao et al. has an obvious defect. Since the entire video clip is sampled at a single sampling rate, the neural network only needs to focus on a few local frames to complete the playback rate prediction. To this end, we introduce a more sophisticated sampling method, that is, the input video clip is composed of multiple video segments with different sampling rates. This design requires the neural network to focus on the input video clip globally. Therefore, our method can obtain a more global video representation which is conducive to applying our method to downstream tasks that require global observation.

For video segment sampling rate prediction tasks, the input of the model is a video clip containing N segments and T video frames, where $s_{v_i}^{(k)} = v_i^j, v_i^{j+m} \ldots, m$ is the sampling rate which means the interval between two adjacent frames in the raw video here. The output of the model is the regression of the sampling rate of each video segment which is indicated as $\alpha \in R^N$. We still use MLP to regress the sampling rate of each video segment. The specific operations are as follows:

$$r_{v_i} = f_\theta(x_{v_i}) \tag{5}$$

$$\alpha = W_1 r_{v_i} + b_1 \tag{6}$$

where r is the video representation extracted by the backbone, W and b are the parameters of the linear classifier, and α is the prediction of the neural network. $\hat{\alpha}$ indicates the ground truth of the sampling rate, and we use the mean square error (MSE) to measure the error between the network prediction and the ground truth:

$$\pounds_{sr} = \frac{1}{n}\sum_{i=1}^{N}(\alpha_i - \hat{\alpha}_i)^2 \tag{7}$$

E. Content-related Contrastive Learning

Both video segment order prediction and sampling rate prediction are only trying to dig the information contained in a video out. However, only considering the information in a video can lead to performance limitations when applying the video representation to downstream tasks. The video representation needs to consider the differences between samples to obtain a better video representation that includes as much task-related information as possible. To achieve this goal, many works introduce contrastive learning into self-supervised video representation learning methods. However, most methods use the correspondence between multiple data streams to set positive and negative samples. For example, an RGB video v_i and its corresponding optical flow o_i are a positive pair, then v_j and o_i are a negative pair. Due to the additional consumption of data calculation and storage, these approaches are not conducive to applying to large-scale video data.

The purpose of contrastive learning is to minimize the distance between positive samples and maximize the distance between negative samples in the feature space. Its essence is to maximize the mutual information of the representations of positive pairs and to make the representations of different samples distribute uniformly in the feature space. We treat video clips sampled from the same video as positive samples and video clips sampled from other videos as negative samples. It can avoid the overhead of additional data calculation and storage by using this method.

For the contrastive learning module, the input video clips are encoded into feature vectors $f_\theta(x)$ that contain rich semantic information by the backbone, then use $h(\cdot)$ to map it into feature space. $x_{v_i}^1$ is a video clip sampled from v_i, and $x_{v_i}^2$ indicates another video clip sampled from the same video. We treat $x_{v_i}^1$ and $x_{v_i}^2$ as a positive pair, $x_{v_i}^1$ and $x_{v_j}^k$ as a negative pair. We denote the features encoded from video clips as $\psi_{x_{v_i}^k} = h(f_\theta(x_{v_i}^k))$, and define $Sim(\psi_{x_{v_i}^1}, \psi_{x_{v_i}^2})$ as

the similarity of feature vector $\psi_{x^1_{v_i}}$ and $\psi_{x^2_{v_i}}$. We maximize the mutual information between $\psi_{x^1_{v_i}}$ and $\psi_{x^2_{v_i}}$ and the dis-tance between $\psi_{x^1_{v_i}}$ and $\psi_{x^1_{v_j}}$ in feature space by minimizing infoNCE [26]. The loss of contrastive learning is as follows:

$$\pounds^{v_i}_{ctr} = -\log \frac{e^{Sim(\psi_{x^1_{v_i}}, \psi_{x^2_{v_i}})}}{e^{Sim(\psi_{x^1_{v_i}}, \psi_{x^2_{v_i}})} + \sum_{j \neq i} e^{Sim(\psi_{x^1_{x_i}}, \psi_{x^1_{v_j}})}} \quad (8)$$

The optimization of the contrastive loss requires a large number of negative samples to avoid the collapse solution, so we maintain a dynamic queue to store the encoded features like moco [40]. The existing methods generally measure the similarity of feature vectors using Euclidean distance or cosine similarity, but they are not suitable for our self-supervised video representation learning method. The reason is that we disrupted the order of the input video segments before performing the order prediction, and the change in order is reflected in the encoded feature vector. Therefore, using these two methods to measure feature vectors can lead to the loss of temporal information. We need to align the features before performing the similarity measurement, otherwise, it may cause the neural network to depend on the appearance features. To this end, we propose a novel feature similarity measurement method for aligning and measuring feature vectors, which is referred to as shuffle similarity.

F. Shuffle Similarity

Given two feature vectors $z_{x_{v_p}}$ and $z_{x_{v_q}}$, we divide it into k parts uniformly, that is $z_{x_v} = z^{(1)}_{x_v}, z^{(2)}_{x_v}, \ldots, z^{(k)}_{x_v}$. Here $z_{x_v} \in R^d$ and d is a positive integer which is divisible by k where k indicates the number of segments that each video clip contained. Then we use Euclidean distance to measure the distance between each part for simplicity as follows:

$$D_{ij} = \sum_{i=1}^{k} \sum_{j=1}^{k} ||z^{(i)}_{x_{v_p}} - z^{(j)}_{x_{v_q}}||_2 \quad (9)$$

To compute the minimum matching distance dis according to the distance matrix, we define $M = \emptyset$, $N = \emptyset$ and $dis = 0$. M is a set storing the minimum matching segment obtained in $z_{x_{v_p}}$, and N is a set storing that in $z_{x_{v_q}}$. Each segment in $z_{x_{v_p}}$ or $z_{x_{v_q}}$ can get the minimum match once. The specific steps are as follows:

$$(m, n) = \arg \min_{m \notin M, n \notin N} D \quad (10)$$

$$dis = dis + D_{mn} \quad (11)$$

$$M = M \cup \{D_{ij} | i = m\}, N = N \cup \{D_{ij} | j = n\} \quad (12)$$

The minimum matching distance of the two feature vectors dis can be obtained after repeating (10)(11)(12) for k times. This measurement approach aligns the feature vectors in segments, so the similarity between the two feature vectors can be reflected robustly and effectively. The pseudo-code is shown in Algorithm 1.

Algorithm 1 Shuffle Similarity

Require:
The feature vector $z_{x_{v_p}} \in R^d$, where $d = n \times k$;
The feature vector $z_{x_{v_q}} \in R^d$, where $d = n \times k$;
Ensure:
The distance between $z_{x_{v_p}}$ and $z_{x_{v_q}}$, dis;
Divide the feature into k parts uniformly, that is $z_{x_v} = z^{(1)}, z^{(2)}, \ldots, z^{(k)}$;
Initialize a distance matrix $D \in R^{k \times k}$;
for each $i \in [1, k]$ **do**
  for each $j \in [1, k]$ **do**
    $D_{ij} = ||z^{(i)}_{x_{v_p}} - z^{(j)}_{x_{v_q}}||_2$;
  end for
end for
Set $M = \emptyset$, $N = \emptyset$ and $dis = 0$;
while $M \neq D$ and $N \neq D$ **do**
  $(m, n) = \arg\min_{m \notin M, n \notin N} D$;
  $dis = dis + D_{m,n}$;
  $M = M \cup \{D_{ij} | i = m\}, N = N \cup \{D_{ij} | j = n\}$;
end while

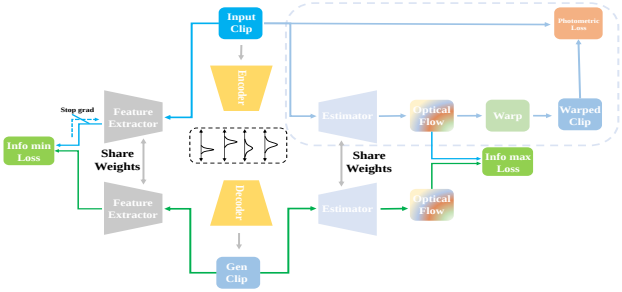


Fig. 3. Given video clip is fed into the autoencoder and optical flow estimator to generate a new view and optical flow. The estimated optical flow and the raw video clip are used to calculate the photometric loss. The info-max loss is obtained by maximizing the similarity between the estimated optical flow of the raw video clip and the generated view. Both the generated view and the raw video clip are fed into the feature extractor to obtain the embeddings, and then the info-min loss is calculated.

G. View Synthesis

Sampling a video clip $x_{v_i} \in R^{C \times T \times H \times W}$ from the raw video, then feed it into the optical flow estimator O_θ and the variational autoencoder G_θ respectively. The estimated optical flow is represented as $o_{x_{v_i}} = O_\theta(x_{v_i})$ and $o_{x_{v_i}} \in R^{2 \times (T-1) \times H \times W}$. These two channels indicate the offset distance of the pixel in the x-axis and y-axis directions, respectively. $o_{x_{v_i}}$ can be used to represent the movement of pixels in x_{v_i}. Then x_{v_i} is used to generate $x^p_{v_i}$ and $x^n_{v_i}$, where $x^p_{v_i} = x^{(1)}_{v_i}, x^{(2)}_{v_i}, ..., x^{(T-1)}_{v_i}$ and $x^n_{v_i} = x^{(2)}_{v_i}, x^{(3)}_{v_i}, ..., x^{(T)}_{v_i}$. To evaluate the effectiveness of the generated optical flow, we utilize the estimated optical flow $o_{x_{v_i}}$ to warp $x^p_{v_i}$ and the warped video clip can be calculated as $x^w_{v_i} = o_{x_{v_i}} x^p_{v_i}$. We set a photometric consistency term that encourages the estimated flow to align video frame patches with a similar appearance by penalizing photometric dissimilarity. The photometric loss is as follows:

$$\pounds_{photo} = \frac{1}{n} \sum \rho(x^n_{v_i} - x^w_{v_i}) \quad (13)$$

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: DOI10.1109/TCSVT.2021.3114209, IEEE Transactions on Circuits and Systems for Video Technology

where $\rho(x) = \sqrt{x^2 + \epsilon^2}$ is the Charbonnier penalty function (a differentiable variant of l_1 norm) [41] and we set $\epsilon = 1e{-}3$. In the pre-training process of the view synthesis module, the parameters of the optical flow estimator are optimized by an independent optimizer. However, many effective conventional optical flow estimation algorithms require a lot of calculation time, which obviously cannot meet the requirements. Therefore, the proposed optical flow estimation module is a rational solution. As shown in Fig. 3, the synthesized view $x^s_{v_i} = G_\theta(x_{v_i})$, $x^s_{v_i} \in R^{C\times T\times H\times W}$. In addition, the Gaussian noise is introduced to improve the generalization of the autoencoder by minimizing the Kullback–Leibler loss which can be calculated as follow:

$$\pounds_{KL-Loss} = \frac{1}{2}\sum_{i=1}^{d}(\mu^2_{(i)} + \sigma^2_{(i)} - \log\sigma^2_{(i)} - 1) \tag{14}$$

Next, we maximize the motion information of the synthesized view and the raw video clip . The synthesized view is also fed into the optical flow estimator to get the optical flow $o^s_{x_{v_i}}$. Then the info-max loss is as below:

$$\pounds_{infomax} = \frac{1}{n}\sum \rho(o_{x_{v_i}} - o^s_{x_{v_i}}) \tag{15}$$

It should be noted that when calculating the info-max loss, $o_{x_{v_i}}$ is separated from the dynamic graph and the parameters of the optical flow estimator are resumed to the original values after the backpropagation. The purpose is to avoid the collapse solution. We minimize the mutual information between the embedding of the synthesized view and the original video clip to suppress the appearance dependence. Then, the info-min loss is calculated as below:

$$\pounds_{infomin} = -D(f_\theta(x_{v_i}), f_\theta(x^s_{v_i})) \tag{16}$$

where we empirically use the cosine similarity as D(.) and f_θ denotes the feature extractor in Fig. 3. In addition, a stop-gradient operation is introduced to avoid the collapse solution. That is x_{v_i} does not contribute to backpropagation.

The pre-training of the view synthesis module is to minimizing $\pounds_{view}$. In the pre-training process of the self-supervised learning, only the parameters of the autoencoder are retained for inference.

H. Learning Framework

One goal of video representation learning under the self-supervised manner is to obtain more generalized and robust video representation, which requires the network to extract semantic information from the video as much as possible. We believe that the high-level semantic information required by the neural network to complete different pretext tasks is not absolutely the same, so it is not easy to ensure that the learned video representation contains enough information to solve multiple downstream tasks. Therefore, we use a multi-task learning framework with hard parameter sharing to optimize multiple pretext tasks jointly. The premise of using this network structure is that the neural network has redundant parameters for the solution of each subtask, otherwise, it may cause some subtasks to become noise affecting the convergence of other subtasks. Besides, the multi-task structure allows each subtask to share parameters, which can effectively prevent a single subtask from falling into overfitting. The parameter sharing mechanism may help the neural network find a better mapping way for completing tasks.

Fig. 2 is the framework of our proposed method. Given a video, we sample a video clip from the raw video and referred to it as $query$. Next fed it into the view synthesis module to generate a new view and resample another video clip as the candidates for key. Then randomly select a sample as the key. After the backbone encodes $query$, the prediction of the order distribution can be obtained through a linear classifier, and the sampling rate prediction can be regressed through another linear classifier. Furthermore, the feature representation of $query$ is mapped into the feature space through a multi-layer non-linear classifier. The feature representation of $query$ forms a positive pair with that of the key. Meanwhile, this maintains a dynamic queue in which many encoded feature representations are stored and these feature representations can be treated as negative samples.

The optimization goal is as follows:

$$minimize\pounds_{total} = \lambda_1\pounds_{order} + \lambda_2\pounds_{sr} + \lambda_3\pounds_{contrastive} \tag{17}$$

where λ_1,λ_2, and λ_3 are hyperparameters, which represent the weight of each subtask that used to balance the contribution of each subtask in backpropagation. Rational weight assignment can avoid easy subtasks dominating the entire training process.

The closest work to the proposed method is the one proposed by Wang et al [12]. In comparison with it, we have the following improvements: (1) Our work extends a video segment order prediction subtask, which can effectively introduce temporal information for video representation. (2) We sample several video segments at different sampling rates and concatenate them into a video clip. The proposed method can generate a more global feature representation. (3) A novel feature vector similarity measurement method is proposed to eliminate appearance dependence to some extent. (4) Different from other contrastive learning based methods [12], [23], [25], [42] that design view transformation based on human prior knowledge, we propose a view synthesis framework for generating effective positive views.

IV. EXPERIMENTS

A. Experimental Setup

We conduct all of the experiments on a server with one Nvidia 2080Ti GPU, one Intel Core i9-9820X CPU, and 64GB RAM under Ubuntu 18.04 LTS and the proposed method is implemented based on Pytorch framework.

B. Datasets

To evaluate the performance of the proposed method, we conduct extensive experiments on multiple datasets. The proposed method is utilized to perform pre-training under the self-supervised manner on the Kinetics-400 [7] and UCF101 [43] datasets, and evaluate the performance of action recognition and video retrieval on UCF101 and HMDB51 [44] datasets.

Kinetics-400 [7] is a large-scale human action recognition dataset containing 400 action categories and 306245 trimmed

videos about ten seconds. Generally, there are $400 \sim 1150$ videos for each action category. We use its training split as the data for pre-training which contains approximately 24000 videos.

UCF101 [43] is one of the most widely used action recognition datasets. It is composed of 5 categories of actions including human-object interaction, body-motion only, human-human interaction, playing musical instruments, and sports. 101 human actions in the natural environment are contained in this dataset, each human action is performed by 25 performers, and each person performs $4 \sim 7$ groups. The video frame size is cropped to 320×240. In this paper, we use the training split of UCF101 as the pre-training data and evaluate the performance of the proposed method in downstream tasks on the testing split.

HMDB51 [44] is a dataset with wide data sources, including movies, some existing released datasets, YouTube videos, etc. It consists of 51 human action categories, and each category contains more than 101 videos. We use this dataset to evaluate the performance of the proposed method in downstream tasks.

C. Backbones

As far as our concern, the evaluation of self-supervised video representation learning methods should be decoupled from the selected network backbone because the performance improvement may come from the use of a powerful network backbone. For a fair comparison, we use the three most widely used network backbones in self-supervised video representation learning methods to evaluate our method, including C3D [1], R3D, and R(2+1)D [2].

C3D [1] is a 3D convolutional neural network expanded from a 2D convolutional neural network, which is suitable for extracting spatio-temporal dynamic information of video by using the 3D convolution kernel. C3D contains 8 stacked convolutional layers and 5 pooling layers, and we set the size of the convolution kernel to $3 \times 3 \times 3$.

R3D [2] is a 3D extended version of 2D ResNet[45]. As we know, ResNet has achieved great success in the image classification field, and it is now widely used in various representation learning approaches. Inspired by it, R3D introduces residual connections in the 3D convolutional neural network to form an 18-layer 3D version of ResNet. The size of the convolution kernel is also set to $3 \times 3 \times 3$.

R(2+1)D [2] decomposes the 3D convolution kernel into a combination of spatial convolution and temporal convolution. The biggest advantage of this mechanism is that the number of R(2+1)D network layers is more than that of normal 3D convolutional neural networks with the same number of spatio-temporal convolution operations, which brings more nonlinear expression capability to the network.

S3D-G [46] adopts a strategy similar to R(2+1)D [2]. On the basis of I3D[3], the 3D convolution kernel is replaced by a combination of 2D + 1D convolution kernels. In addition, the feature gating module is introduced to generate the weight of the channel dimension.

D. Pretext Task Training

As shown in Fig. 2, the non-linear classifier is connected to the network backbone during the pretext task training stage and it is removed when evaluating downstream tasks. Perceiving long-term video features is helpful for completing most video understanding tasks, so the length of the input video clip has a very significant impact on the performance of downstream tasks. However, the benefits of increasing the length of video clips are not the focus of this paper, so we use the same configuration as the recent works [9], [11] that limiting the length of video clips to 16 frames for the sake of fairness. The sampling process is shown in Fig. 1, except that the index of the first frame is randomly selected to ensure that each sampled video clip is unique. For the data augmentation, the input video clip is first resized to 128×171, and then randomly crop it into 112×112, then color jitter is used to randomly adjust the brightness, contrast, and saturation of the input video clip, and finally randomly horizontal flip the video clip. When pre-training on the UCF101 [43] dataset, the batch size is set to 16 and train for 300 epochs. The batch size and the total number of epoch are setted to 32 and 30 respectively when pre-training on Kinetics-400 [7] dataset. We select SGD as the optimizer, the initial learning rate is set to $1e-3$ while the momentum and weight decay are set to $9e-1$ and $5e-4$ respectively. It is noticed that the learning rate reduces to 0.1 times after 200 (20 for Kinetics) epochs pre-training.

It can be seen from Fig. 2 (b) that the video clip $query$ and key are encoded into 512-dimensional feature vectors by the encoder Q and the encoder K, and the features are denoted as v_q and v_k here. v_q is used for order prediction (d) and sampling rate prediction (c), while v_q and v_k are mapped into 128-dimensional feature vectors respectively through a non-linear classifier to perform contrastive learning with the negative samples stored in the queue (e). It is worth noting that we use an engineering method similar to moco [40]. The encoder K does not perform backpropagation during the pre-training stage, and the encoder K updates the parameters in a momentum update way. For each iteration, the encoder K updates the parameters in the following manner:

$$\theta_k = \delta\theta_k + (1-\delta)\theta_q \tag{18}$$

where θ_k is the parameters of the encoder K, and θ_q represents the parameters of the encoder Q. For ease of implementation, the length of the queue in this project is set to the integer multiple of the batch size. The proposed method is a multi-task structure, we assign different weights to the loss of each subtask to control the training speed of each subtask. In our experiment, $\pounds_{total} = \lambda_1\pounds_{order} + \lambda_2\pounds_{sr} + \lambda_3\pounds_{ctr}$, where $\lambda_1 = \lambda_3 = 1$ and $\lambda_2 = 0.3$.

E. Fine-tune and Evaluation

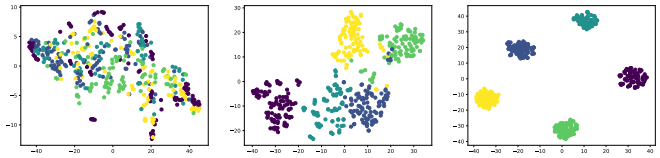
To evaluate the superiority of the proposed method, we apply the network backbone trained by our proposed method to the task of action recognition. Specifically, all parameters of the convolutional layer in the backbone after self-supervised learning are retained, and the parameters of the fully connected layer are randomly initialized. During the fine-tune stage, the

training set of the Kinetics-400 [7] and UCF101[43] are used as the training data. It should be noted that the video clip fed into the neural network is continuous 16 frames that randomly sampled from the raw video. For data augmentation, it is consistent with the configuration of the pretext task training.

We evaluate the performance of the learned video representations applied to downstream tasks on the testing sets of UCF101 [43] and HMDB51 [44]. We firstly sample 10 clips in a video, then resize the video frame to the size of 128×171, and center crop it into the size of 112×112. The final prediction of the classification is obtained by averaging the prediction of each clip.

F. Ablation Study

In this section, we conduct extensive ablation experiments to further evaluate the superiority of our proposed method. The following three aspects are discussed. Firstly, we discuss why self-supervised learning works. Secondly, the best experimental setup is explored. Then, experiments are conducted to explore the effectiveness of each component. Finally, experimental results are analyzed and discussed.



(a) Random (b) Self-supervised (c) Supervised

Fig. 4. Visualization of feature distribution. Each color in the figure represents an action category. Here are 5 colors in total, representing class 0, class 10, class 20, class 30, and class 40 of the UCF101[43] dataset respectively. (a) The distribution of the features extracted by the backbone whose parameters are randomly initialized. (b) The distribution of features extracted by the backbone after self-supervised learning. (c) Features distribution that the features are extracted by the backbone after supervised learning.

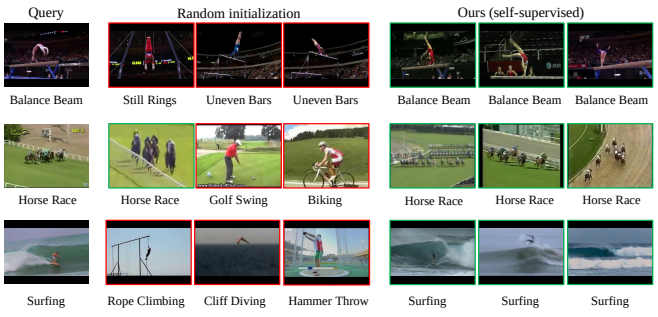


Fig. 5. Each image represents a video. From left to right, the first column represents three querys randomly selected from the testing split. The following are the Top-3 retrieval results of using the backbone with random initialization parameters and the backbone pre-trained by the proposed method. The subscript of each image represents the label it belongs to. The correct retrieval results are marked in green, and failure cases are marked in red.

1) ***Why self-supervised learning works****:* We are eager to know why self-supervised learning works. In this part, we use R(2+1)D as backbone and sample video clips which consists of 16 consecutive frames from 5 action categories in UCF101 dataset. Each video clip can be represented by a 512-dimensional feature vector after encoded by the network backbone. Then, t-SNE [47] is used to map these feature vectors from high-dimensional space to a two-dimensional feature space.

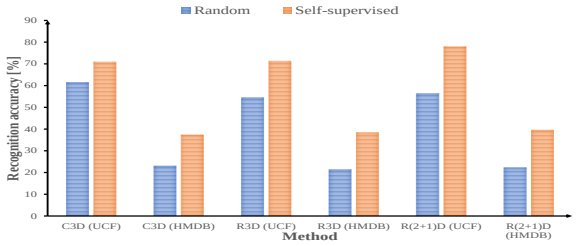


Fig. 6. Comparing the performance of action recognition on three backbones using different parameter initialization methods. Blue indicates that the parameters of the backbone are initialized randomly, while orange indicates that the parameters of the backbone are trained by the proposed method. The vertical axis represents the accuracy of action recognition.

From Fig. 4(a), it can clearly be observed that the features extracted distribute in the entire feature space uniformly when the parameters of the neural network are initialized randomly. Self-supervised learning introduces much temporal and spatial semantic information into the feature representation. This semantic information makes the extracted feature representation more discriminative. As shown in Fig. 4(b), the points with the same color (representing the same action category) in the figure are more concentrated in the feature space. It should be noted that we do not introduce any manual annotation during the self-supervised learning stage. Fig. 4(c) shows the feature distribution after supervised learning. It can be seen that benefit from many action category labels, the point distribution of the same category is extremely concentrated, and the distance between different categories is very far. It can also be seen from Fig. 5 that when the network parameters are randomly initialized, the extracted features can almost only reflect the background color of the video. However, after the pre-training of the self-supervised method, rich semantic information is introduced into the feature representation, which makes the retrieval result significantly better than the former.

TABLE II
USING THREE BACKBONES TO EVALUATE THE PERFORMANCE OF RANDOM INITIALIZATION AND SELF-SUPERVISED LEARNING ON ACTION RECOGNITION TASKS ON TWO DATASETS.

Backbone	Eval. dataset	Random	Self-supervised
C3D	UCF101	61.5	**75.9**
C3D	HMDB51	23.1	**39.8**
R3D	UCF101	54.5	**76.6**
R3D	HMDB51	21.4	**42.3**
R(2+1)D	UCF101	56.4	**80.1**
R(2+1)D	HMDB51	22.3	**46.6**

It can be seen from Table II that with C3D [1] backbone, the proposed approach can obtain 14.4% and 16.7% improvements on UCF101 [43] and HMDB51 [44] than randomly initialized. With R3D [2] as a backbone, it can achieve 22.1% and 20.9% improvements, respectively. And when we use R(2+1)D [2] as a backbone, the improvements can be 23.7% and 24.3%.

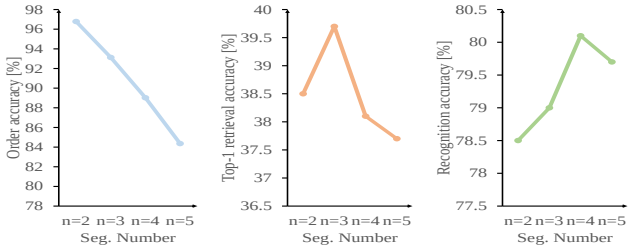


Fig. 7. The order prediction accuracy, top-1 video retrieval accuracy and action recognition accuracy on UCF-101 dataset with the different video segment numbers. The blue line chart represents the relationship between the order prediction accuracy and the number of video segments, while the orange and green line chart respectively represent the relationship between the top-1 retrieval accuracy and the action recognition accuracy and the number of video segments.

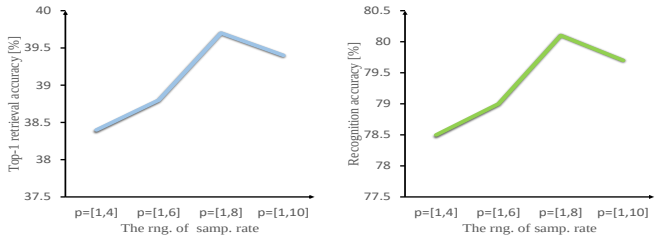


Fig. 8. The top-1 video retrieval accuracy and action recognition accuracy on UCF-101 dataset with the different ranges of the candidate sampling rate. p denotes the range of the candidate sampling rate. The blue line chart represents the relationship between the top-1 retrieval accuracy and the range of the candidate sampling rate while the green line chart represents the relationship between the action recognition accuracy and the range of the candidate sampling rate.

Another key observation can be found from Fig. 6 is that the improvement of the proposed method on the hmdb51 [44] is always greater than that on the UCF101 [43]. This reflects that when the proposed method is used as a pre-training method, the benefits on small datasets are greater than those on large datasets.

The above experimental results strongly demonstrate that when the neural network pre-trained by self-supervised learning is used as a feature extractor and applied to downstream tasks, the neural network can extract discriminative video representations without any manual annotation. When the proposed method is regarded as a pre-training method, it can help the network find a better mapping way to complete the target task.

TABLE III
COMPARING THE IMPACT OF THE DIFFERENT NUMBERS OF VIDEO SEGMENT ON THE TOP-1 RETRIEVAL RESULT AND THE ACTION RECOGNITION ACCURACY

Method	Order acc.	Samp. error	Cntr. loss	Top-1	Recg. acc.
$n = 2$	96.78	1.55	0.72	38.5	78.8
$n = 3$	93.12	1.66	0.66	**39.7**	**80.1**
$n = 4$	89.02	1.88	0.69	38.1	79.0
$n = 5$	84.36	2.23	0.75	37.7	78.3

n: the number of video segment. Samp. error: the error between sampling rate prediction and ground truth. Order acc: the accuracy of the order prediction. Cntr. loss: the loss of contrastive learning. Top-1: the top-1 retrieval result. Recg. acc: the action recognition accuracy.

*2) **Number of Video Segments**:* We explore the best video segment number for pretext task with R(2+1)D [2] as the backbone in Table III. When the input video clip contains n segments, the order prediction task becomes a classifier problem with $n!$ classes. However, too many segments can result in fewer frames per segment, which is not conducive to sampling rate prediction. We limit the number of segments to $n = [2, 5]$ for the comparative experiment to control the difficulty of the pretext task. It can be seen from the table that the performance of downstream tasks increases with the increase of the number of segments until $n = 3$. When $n > 3$, the accuracy of downstream tasks begins to decline. From Fig. 7, we can see that the accuracy of the order prediction decreases as the number of video segments increases. This reflects that the difficulty of the pretext task increases with the number of segments increases. The results prove the hypothesis that the pretext designed should not be too difficult or too simple. Also, it can be seen from Fig. 7 that the performance of action recognition and the performance of video retrieval are positively correlated. The reason might be that there is a lot of relevant information between these two tasks.

TABLE IV
COMPARING THE IMPACT OF THE DIFFERENT RANGES OF CANDIDATE SAMPLING RATE ON THE TOP-1 RETRIEVAL RESULT AND THE ACTION RECOGNITION ACCURACY

Method	Order acc.	Samp. error	Cntr. loss	Top-1	Recg. acc.
$p = [1, 4]$	90.89	2.23	0.75	38.4	78.5
$p = [1, 6]$	91.34	2.01	0.68	38.8	79.0
$p = [1, 8]$	90.77	1.73	0.72	**39.7**	**80.1**
$p = [1, 10]$	91.25	1.68	0.70	39.4	79.7

p: the range of candidate sampling rate. Samp. error: the error of sampling rate prediction. Order acc: the accuracy of the order prediction. Cntr. loss: the loss of contrastive learning. Top-1: the top-1 retrieval result. Recg. acc: the action recognition accuracy.

TABLE V
THE COMPARISON OF MODELING THE SAMPLING RATE PREDICTION AS A CLASSIFICATION PROBLEM AND A REGRESSION PROBLEM ON UCF101

Method	Backbone	Top-1	Recog. acc.
Classification	C3D	34.7	74.5
Regression	C3D	35.6	75.9
Classification	R(2+1)D	38.6	79.3
Regression	R(2+1)D	39.7	80.1

*3) **Sampling Rate**:* During the training process of pretext tasks, except for the number of the video segments, we also explore the impact of different sampling rate ranges on downstream tasks. In this part, we discuss the optimal range of sampling rate $p \in [d, u]$.

From the Table IV, we can see that 4 groups of experiments are performed, $p = [1, 4]$, $p = [1, 6]$, $p = [1, 8]$ and $p = [1, 10]$. p represents the interval between two adjacent frames in the sampled video clip. When the candidate range of the sampling rate is too small, the change of adjacent frames is not obvious enough, it may limit the performance of the proposed method on downstream tasks.

In addition, we also discuss the advantages of modeling the motion amplitude prediction of moving objects as a regression problem compared to a classification problem in Table V. The table shows the superiority of modeling sampling rate prediction as a regression problem.

TABLE VI
EXPLORING THE EFFECTIVENESS OF EACH COMPONENT

Components					Downstream task	
Order pred.	Samp. rate pred.	Cntr. learning	Shuffle	View	Top-1	Recog. acc.
✓	✗	✗	✗	✗	10.2	73.5
✓	✓	✗	✗	✗	11.4	75.6
✓	✓	✓	✗	✗	38.0	77.2
✓	✓	✓	✓	✗	38.5	77.8
✓	✓	✓	✓	✓	39.7	80.1

Order pred: the order prediction subtask; Samp. rate pred: the sampling rate prediction subtask; Cntr. learning: the contrastive learning module; Top-1: the top-1 retrieval result on UCF101 dataset; Recg. acc: the action recognition accuracy on UCF101 dataset.

4) Effectiveness of Resolution and Length: Table IX shows the performance of the proposed method on video clips under different resolutions and lengths. Due to equipment limitations, we use the relatively lightweight networks R3D [2] and S3D-G [46] as the backbone for this part of the experiment. From the table, we have the two very intuitive observations: (1) Higher resolution produces better results. (2) More video frames are fed, better performance can be achieved. This shows that adding data is still an effective way to improve the performance of the target task. However, the increment in data can also significantly increase the computational cost.

*5) **Effectiveness of Each Component**:* In this part, we discuss the effectiveness of each component of the proposed approach. We select R(2+1)D [2] as the backbone, use the train split of UCF101 [43] as the pre-training data, then fine-tune the network on the training set of UCF101 and HMDB51, and evaluate the effectiveness of each component on their test split.

According to the content in Table VI, we have the following key observations: (1) When only performing order prediction and sampling rate prediction subtasks, the considerable performance can be achieved. (2) The introduction of contrastive learning significantly improves the performance of video retrieval, which shows that contrastive learning can significantly enhance the discrimination of video representation. (3) Shuffle Similarity can further improve the performance of our proposed method on downstream tasks.

TABLE VII
THE PRETEXT TASK PERFORMANCE OF SINGLE-TASK AND MULTI-TASK ON UCF101

Backbone	Order acc.	Samp. error	Cntr. loss
R(2+1)D	94.16	—	—
R(2+1)D	—	1.58	—
R(2+1)D	—	—	0.50
R(2+1)D	93.75	1.63	0.54

TABLE VIII
THE COMPARISON WITH VCOP[9]

Method	Backbone	FLOPs	Recog. acc.	Top-1
VCOP[9]	C3D	115.8×10^9	65.6	7.4
Ours(order)	C3D	38.6×10^9	66.7	7.8
VCOP[9]	R3D	59.8×10^9	64.9	14.1
Ours(order)	R3D	19.9×10^9	65.3	14.5
VCOP[9]	R(2+1)D	64.4×10^9	72.4	10.7
Ours(order)	R(2+1)D	21.5×10^9	73.5	10.5

TABLE IX
THE IMPACT OF DIFFERENT FRAME RESOLUTIONS AND CLIP LENGTHS ON THE DOWNSTREAM TASK

Backbone	Resolution	Clip length	UCF101	HMDB51
R3D	112×112	16	76.6	42.3
R3D	112×112	32	78.4	44.5
R3D	112×112	64	80.2	46.0
R3D	152×152	16	78.9	43.8
R3D	224×224	16	81.0	46.6
S3D-G	112×112	16	70.6	39.1
S3D-G	112×112	32	74.8	42.7
S3D-G	112×112	64	79.5	47.3
S3D-G	152×152	16	74.9	43.6
S3D-G	224×224	16	78.6	46.5

To further demonstrate the superiority of the proposed method, we show the comparison between the proposed method and VCOP[9] a method based on video order prediction in Table VIII. The results in Table VIII are performed with the proposed method only considering the video segment order prediction subtask. It can be seen that even if only the video segment order prediction subtask is considered, the proposed method can still achieve slightly better results than VCOP[9], and only costs about one-third of the calculation overhead.

Our proposed method utilizes a multi-task learning architecture to optimize three subtasks together. Therefore, we need to ensure that the network has enough parameters to optimize multiple subtasks jointly. Table VII shows the results of the ablation experiments. It can be seen that using a multi-task learning structure to optimize three subtasks at the same time can achieve a comparable result to single-task optimization. This strongly proves that the network does have enough parameters to jointly optimize multiple subtasks.

G. Discussion

By visualizing the distribution of the features extracted by the network in the feature space, we find that a considerable clustering result can be achieved even without any manual annotation is introduced. This phenomenon strongly illustrates that the proposed method can fully dig out the temporal and spatial semantic information hidden in the video, and then make the learned video representation more discriminative. Compared to random initialization, the proposed method as a parameter initialization method not only converges faster but also greatly surpasses the performance of the former. This benefits from much temporal and spatial semantic information obtained by pre-training and the designed pseudo-labels can drive the network to extract more meaningful features.

Also, we perform several groups of ablation experiments to explore the influence of different video segment numbers, the ranges of candidate sampling rate, and each component of the proposed method on downstream tasks. Experimental results demonstrate that a moderately difficult pretext task is essential

for learning a good video representation, and its impact is even more significant than that of other pre-training methods.

Although the proposed method has satisfactory performance in downstream tasks and various ablation experiments also show that the video representation learned by the proposed method can effectively reflect the dynamic changes of the video. However, this method still has some limitations. For example, the motion speed of different types of actions are different, that is, the same sampling rate of different actions may be accompanied by different motion amplitudes. Therefore, directly using the sampling rate as a metric for the motion amplitude of the moving objects is sometimes not accurate. Recent self-supervised video representation learning works suggest that the premise for self-supervised learning to be effective is that the designed pretext tasks and target downstream tasks must contain enough task-related information. Although the existing self-supervised video representation learning methods have made encouraging progress, these methods are designed based on the prior knowledge of researchers. However, searching for the best pretext tasks is still a challenging task. Another key observation of this work is that compared to a single-task learning framework, optimizing multiple pretext tasks at the same time can help the network dig out more temporal and spatial semantic information. Therefore, after the exploration of this paper, we have the following insights: if the video representation is learned for a specific downstream task, the optimal pretext task is often difficult to find, so the guidance of a small number of labels may be a better solution. If the generalization of video representation is pursued, it can be more effective to optimize multiple pretext tasks with suitable difficulty jointly.

V. Comparison with state-of-the-art methods

A. Comparison Methods

First, we briefly introduce the comparison methods in this section. VCOP[9] is a self-supervised video representation learning method based on video clip order prediction. VCP[10] attempts to drive the network to extract video semantic information in the process of classifying the type of transformation. ST-Puzzle[36] extends the image puzzles into three-dimensional space-time cubic puzzles. PRP[11] firstly uses temporal resolution perception to learn video representation. The pace[12] proposed by Wang et al. also introduces contrastive learning on the basis of temporal resolution perception, which further improves the performance.

B. Action Recognition

We compare our proposed approach with other state-of-the-art methods for action recognition. The comparison results are shown in Table X. The results in the table clearly show that the proposed method can achieve state-of-the-art performance on both UCF101 and HMDB51 dataset with different backbones. When C3D is used as the backbone, the proposed method outperforms the current best-performing method RTT [13] by 7.6% on UCF101 [43]. When we use R3D [2] as backbone, the proposed method outperforms PRP[11] by 10.1% on UCF101 and 12.6% on HMDB51 respectively. Besides, even with lower resolution and pre-training on the smaller dataset, the proposed method still outperforms DPC [29] by 8.4% and 7.8% on UCF101 and HMDB51 datasets. When using R(2+1)D [2] as the backbone and pre-training on the UCF101 dataset, the proposed method outperforms the current state-of-the-art method of Pace [12] by 4.2% and 10.7%. Even if Pace [12] uses larger dataset kinetics [7] for pre-training, the proposed method still outperforms it by 3.0% and 10.0%. Due to equipment limitations, when the S3D-G[46] is used as the backbone and the input video clip length is 64, we can only execute experiments with a lower video resolution of 152×152. Nevertheless, the proposed method can still achieve comparable performances in comparison with the current state-of-the-art method in [12].

C. Video Retrieval

We further verify the superiority of the proposed method on the nearest neighbour video retrieval task. For a fair comparison, we follow the evaluation protocol used in [9]. First sample 10 16-frames video clips from each video in the training set, and then feed them into the backbone to obtain 512-dimensional feature vectors from the last pooling layer. Then extract the features of the video clips from the testing set to query the clips in the training set. The specific operation is that we use the cosine similarity to measure the distance between the feature representation of the query clip and the feature representation of all clips in the training set to obtain k nearest neighbor video clips. If the label of the test clip is within the $Top-k$ retrieval results range, it means that this retrieval is successful.

The comparison results on UCF101 and HMDB51 datasets are respectively shown in Table XI and Table XII. The tables show top-1, top-5, top-10, top-20 and top-50 retrieval accuracy. It can be seen that the proposed method outperforms to the listed state-of-the-art methods in all evaluation metrics. In addition, we can also see that applying the pre-trained parameters to video retrieval can achieve a more significant performance improvement than action recognition. From Table VI, we can find that the huge improvement in video retrieval comes from the introduction of contrastive learning. For the proposed method, the order prediction subtask and the video segment sampling rate prediction subtask are mainly used to introduce semantic information to the network and help the network find better mapping when performing downstream tasks while contrastive learning can significantly enhance the ability of the network to extract discriminative features.

VI. Conclusion

In this paper, a novel self-supervised video representation learning method is proposed, which mines the temporal and spatial semantic information hidden in the video by regressing the sampling rate of the video segments and classifying the order of video segments. In addition, we introduced the currently very popular contrastive learning to further enhance the discrimination of learned video representations. To verify the effectiveness of the proposed method, we use C3D, R3D, R(2+1)D and S3D-G as backbone in two downstream tasks

TABLE X
COMPARISION WITH THE STATE-OF-THE-ART SELF-SUPERVISED LEARNING METHODS FOR ACTION RECOGNITION

Method	Self-supervised learning setup					Action Recognition	
	Backbone	Params	Frame size	Clip length	Pre-training	UCF101	HMDB51
VCOP[9]	C3D	27.7M	112×112	16	UCF101	65.6	28.4
VCP[10]	C3D	27.7M	112×112	16	UCF101	68.5	32.5
ST-Puzzle[36]	C3D	65.5M	224×224	128	Kinetics-400	65.8	33.7
PRP[11]	C3D	27.7M	112×112	16	UCF101	69.1	34.5
RTT[13]	C3D	27.7M	112×112	16	UCF101	68.3	38.4
Ours	C3D	29.1M	112×112	16	UCF101	70.9	37.3
Ours(view)	C3D	32.5M	112×112	16	UCF101	**75.9**	**39.8**
VCOP[9]	R3D	14.2M	112×112	16	UCF101	64.9	29.5
VCP[10]	R3D	14.2M	112×112	16	UCF101	66.0	31.5
ST-Puzzle[36]	R3D	33.6M	224×224	128	Kinetics-400	65.8	33.7
PRP[11]	R3D	14.2M	112×112	16	UCF101	66.5	29.7
3D-RoNet[48]	R3D	33.6M	224×224	16	Kinetics-400	62.9	33.7
DPC[29]	R3D	14.2M	128×128	40	Kinetics-400	68.2	34.5
Ours	R3D	15.0M	112×112	16	UCF101	71.1	38.4
Ours	R3D	15.0M	112×112	16	Kinetics-400	72.1	39.2
Ours(view)	R3D	22.2M	112×112	16	UCF101	**76.6**	**42.3**
VCOP[9]	R(2+1)D	14.4M	112×112	16	UCF101	72.4	30.9
VCP[10]	R(2+1)D	14.4M	112×112	16	UCF101	66.3	32.2
PRP[11]	R(2+1)D	14.4M	112×112	16	UCF101	72.1	35.0
Pace[12]	R(2+1)D	14.4M	112×112	16	UCF101	75.9	35.9
Pace[12]	R(2+1)D	14.4M	112×112	16	Kinetics-400	77.1	36.6
Ours	R(2+1)D	15.2M	112×112	16	UCF101	77.8	39.5
Ours	R(2+1)D	15.2M	112×112	16	Kinetics-400	78.8	42.2
Ours(view)	R(2+1)D	22.5M	112×112	16	UCF101	**80.1**	**46.6**
SpeedNet[49]	S3D-G	9.6M	224×224	64	Kinetics-400	81.1	48.8
Pace[12]	S3D-G	9.6M	224×224	64	UCF101	**87.1**	52.6
Ours	S3D-G	10.1M	152×152	64	UCF101	80.2	50.3
Ours(view)	S3D-G	15.0M	152×152	64	UCF101	85.9	**52.7**

TABLE XI
COMPARISON WITH STATE-OF-THEART METHODS FOR VIDEO RETRIEVAL TASK ON UCF101 DATASET.

Backbone	Method	Top1	Top5	Top10	Top20	Top50
C3D[1]	Random	16.7	28.5	33.5	40.0	49.4
	VCOP[9]	12.5	29.0	39.0	50.6	66.9
	VCP[10]	17.3	31.5	42.0	52.6	67.7
	PRP[11]	23.2	38.1	46.0	55.7	68.4
	Pace[12]	20.0	37.4	46.9	58.5	73.1
	Ours	**35.6**	**52.4**	**62.2**	**73.2**	**83.5**
R3D[2]	Random	9.9	18.9	26.0	35.5	51.9
	VCOP[9]	14.1	30.3	40.4	51.1	66.5
	VCP[10]	18.6	33.6	42.5	53.5	68.1
	PRP[11]	22.8	38.5	46.7	55.2	69.1
	Pace[12]	19.9	36.2	46.1	55.6	69.2
	Ours	**37.1**	**55.2**	**61.7**	**72.8**	**84.2**
R(2+1)D[2]	Random	10.6	20.7	27.4	37.4	53.1
	VCOP[9]	10.7	25.9	35.4	47.3	63.9
	VCP[10]	19.9	33.7	42.0	50.5	64.4
	PRP[11]	20.3	34.0	41.9	51.7	64.2
	Pace[12]	17.9	34.3	44.6	55.5	72.0
	Ours	**39.7**	**53.6**	**66.7**	**75.9**	**83.7**

TABLE XII
COMPARISON WITH STATE-OF-THEART METHODS FOR VIDEO RETRIEVAL TASK ON HMDB51 DATASET.

Backbone	Method	Top1	Top5	Top10	Top20	Top50
C3D[1]	Random	7.4	20.5	31.9	44.5	66.3
	VCOP[9]	7.4	22.6	34.4	48.5	70.1
	VCP[10]	7.8	23.8	35.5	49.3	71.6
	PRP[11]	10.5	27.2	40.4	56.2	75.9
	Pace[12]	8.0	25.2	37.8	54.4	77.5
	Ours	**15.3**	**35.9**	**48.1**	**62.0**	**80.1**
R3D[2]	Random	6.7	18.3	28.3	43.1	67.9
	VCOP[9]	7.6	22.9	34.4	48.8	68.9
	VCP[10]	7.6	24.4	36.6	53.6	76.4
	PRP[11]	8.2	25.8	38.5	53.3	75.9
	Pace[12]	8.2	24.2	37.3	53.3	74.5
	Ours	**15.7**	**40.3**	**53.2**	**66.8**	**82.3**
R(2+1)D[2]	Random	4.5	14.8	23.4	38.9	63.0
	VCOP[9]	5.7	19.5	30.7	45.8	67.0
	VCP[10]	6.7	21.3	32.7	49.2	73.3
	PRP[11]	8.2	25.3	36.2	51.0	73.0
	Pace[12]	10.1	24.6	37.6	54.4	77.1
	Ours	**17.7**	**46.6**	**55.4**	**68.6**	**83.0**

action recognition and video retrieval, respectively. The proposed method can pre-train the backbone as a feature extractor for downstream tasks and can further fine-tune the pre-trained backbone to perform downstream tasks. The experimental results show that our method outperforms existing state-of-the-art self-supervised video representation learning methods on multiple datasets.

REFERENCES

[1] 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.

[2] 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.

[3] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," *Advances in neural information processing systems*, vol. 27, pp. 568–576, 2014.

[4] Y. Li, B. Ji, X. Shi, J. Zhang, B. Kang, and L. Wang, "Tea: Temporal excitation and aggregation for action recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 909–918.

[5] J. Lin, C. Gan, and S. Han, "Temporal shift module for efficient video understanding. 2019 ieee," in *CVF International Conference on*

Computer Vision (ICCV), 2019, pp. 7082–7092.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] D. Luo, C. Liu, Y. Zhou, D. Yang, C. Ma, Q. Ye, and W. Wang, "Video cloze procedure for self-supervised spatio-temporal learning," *arXiv preprint arXiv:2001.00294*, 2020.

[11] 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/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6548–6557.

[12] J. Wang, J. Jiao, and Y.-H. Liu, "Self-supervised video representation learning by pace prediction," in *European Conference on Computer Vision*. Springer, 2020, pp. 504–521.

[13] S. Jenni, G. Meishvili, and P. Favaro, "Video representation learning by recognizing temporal transformations," *arXiv preprint arXiv:2007.10730*, 2020.

[14] A. Piergiovanni, A. Angelova, and M. S. Ryoo, "Evolving losses for unsupervised video representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 133–142.

[15] J. Wang, Y. Gao, K. Li, X. Jiang, X. Guo, R. Ji, and X. Sun, "Enhancing unsupervised video representation learning by decoupling the scene and the motion," *arXiv preprint arXiv:2009.05757*, 2020.

[16] 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.

[17] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," *arXiv preprint arXiv:1803.07728*, 2018.

[18] 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.

[19] 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.

[20] D. Wei, J. J. Lim, A. Zisserman, and W. T. Freeman, "Learning and using the arrow of time," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8052–8060.

[21] N. Sayed, B. Brattoli, and B. Ommer, "Cross and learn: Cross-modal self-supervision," in *German Conference on Pattern Recognition*. Springer, 2018, pp. 228–243.

[22] A. Mahendran, J. Thewlis, and A. Vedaldi, "Cross pixel optical-flow similarity for self-supervised learning," in *Asian Conference on Computer Vision*. Springer, 2018, pp. 99–116.

[23] L. Tao, X. Wang, and T. Yamasaki, "Self-supervised video representation learning using inter-intra contrastive framework," in *Proceedings of the 28th ACM International Conference on Multimedia*, 2020, pp. 2193–2201.

[24] 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.

[25] T. Han, W. Xie, and A. Zisserman, "Self-supervised co-training for video representation learning," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[26] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018.

[27] C. Vondrick, H. Pirsiavash, and A. Torralba, "Generating videos with scene dynamics," in *Advances in neural information processing systems*, 2016, pp. 613–621.

[28] C. Vondrick, A. Shrivastava, A. Fathi, S. Guadarrama, and K. Murphy, "Tracking emerges by colorizing videos," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 391–408.

[29] T. Han, W. Xie, and A. Zisserman, "Video representation learning by dense predictive coding," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2019, pp. 0–0.

[30] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised learning of video representations using lstms," in *International conference on machine learning*, 2015, pp. 843–852.

[31] P. Xu, Z. Song, Q. Yin, Y.-Z. Song, and L. Wang, "Deep self-supervised representation learning for free-hand sketch," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 4, pp. 1503–1513, 2020.

[32] S. Guo, E. Rigall, L. Qi, X. Dong, H. Li, and J. Dong, "Graph-based cnns with self-supervised module for 3d hand pose estimation from monocular rgb," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 31, no. 4, pp. 1514–1525, 2020.

[33] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "Slowfast networks for video recognition," in *Proceedings of the IEEE international conference on computer vision*, 2019, pp. 6202–6211.

[34] H. Cho, T. Kim, H. J. Chang, and W. Hwang, "Self-supervised spatio-temporal representation learning using variable playback speed prediction," *arXiv preprint arXiv:2003.02692*, 2020.

[35] 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.

[36] 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.

[37] N. Chesneau, K. Alahari, and C. Schmid, "Learning from web videos for event classification," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 10, pp. 3019–3029, 2017.

[38] C. Gan, B. Gong, K. Liu, H. Su, and L. J. Guibas, "Geometry guided convolutional neural networks for self-supervised video representation learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5589–5597.

[39] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[40] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9729–9738.

[41] A. Bruhn, J. Weickert, and C. Schnörr, "Lucas/kanade meets horn/schunck: Combining local and global optic flow methods," *International journal of computer vision*, vol. 61, no. 3, pp. 211–231, 2005.

[42] T. Pan, Y. Song, T. Yang, W. Jiang, and W. Liu, "Videomoco: Contrastive video representation learning with temporally adversarial examples," *arXiv preprint arXiv:2103.05905*, 2021.

[43] 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.

[44] 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.

[45] 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.

[46] S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy, "Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 305–321.

[47] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[48] L. Jing, X. Yang, J. Liu, and Y. Tian, "Self-supervised spatiotemporal feature learning via video rotation prediction," *arXiv preprint arXiv:1811.11387*, 2018.

[49] 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/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9922–9931.

Jing Huang is currently pursuing the graduation degree with the Department of Computer Science and Technology, Xiamen University, Fujian, China. His research interests include deep learning, action recognition, self-supervised learning and person re-identification.

Yan Huang is currently pursuing the graduation degree with the Department of Computer Science and Technology, Xiamen University, Fujian, China. Her current research interests include machine learning and computer vision.

Qicong Wang received the Ph.D. degree in information and communication engineering from Zhejiang University, Hangzhou, China. He is currently an Associate Professor at the Department of Computer Science and Technology, Xiamen University, Xiamen, China. His research interests include computer vision, machine learning, big data analytic.

Wenming Yang (Senior Member, IEEE) received his Ph.D. degree in information and communication engineering from Zhejiang University in 2006. He is an associate professor in Shenzhen International Graduate School/Department of Electronic Engineering, Tsinghua University. His research interests include image processing, pattern recognition, computer vision and AI in medicine.

Hongying Meng (M'10-SM'17) received his Ph.D. degree in Communication and Electronic Systems from Xi'an Jiaotong University, Xi'an China. He is an associate editor for IEEE Transactions on Circuits and Systems for Videos Technology (TCSVT) and IEEE Transactions on Cognitive and Developmental Systems (TCDS). He is a Fellow of The Higher Education Academy (HEA) and a member of Engineering Professors Council in UK. He has authored over 130 publications including IEEE TIP, TCYB, TFS, TAC, TCSVT, TBE, TCDS, ICASSP and CVPR. He is currently a reader at the Department of Electronic and Computer Engineering, Brunel University London, U.K. His research interests include digital signal processing, affective computing, machine learning, human computer interaction, and computer vision.