TCGL: Temporal Contrastive Graph for Self-supervised Video Representation Learning

Yang Liu, *Member, IEEE*, Keze Wang, Lingbo Liu, Haoyuan Lan, and Liang Lin, *Senior Member, IEEE*

***Abstract*—Video self-supervised learning is a challenging task, which requires significant expressive power from the model to leverage rich spatial-temporal knowledge and generate effective supervisory signals from large amounts of unlabeled videos. However, existing methods fail to increase the temporal diversity of unlabeled videos and ignore elaborately modeling multi-scale temporal dependencies in an explicit way. To overcome these limitations, we take advantage of the multi-scale temporal dependencies within videos and proposes a novel video self-supervised learning framework named Temporal Contrastive Graph Learning (TCGL), which jointly models the inter-snippet and intra-snippet temporal dependencies for temporal representation learning with a hybrid graph contrastive learning strategy. Specifically, a Spatial-Temporal Knowledge Discovering (STKD) module is first introduced to extract motion-enhanced spatial-temporal representations from videos based on the frequency domain analysis of discrete cosine transform. To explicitly model multi-scale temporal dependencies of unlabeled videos, our TCGL integrates the prior knowledge about the frame and snippet orders into graph structures, i.e., the intra-/inter- snippet Temporal Contrastive Graphs (TCG). Then, specific contrastive learning modules are designed to maximize the agreement between nodes in different graph views. To generate supervisory signals for unlabeled videos, we introduce an Adaptive Snippet Order Prediction (ASOP) module which leverages the relational knowledge among video snippets to learn the global context representation and recalibrate the channel-wise features adaptively. Experimental results demonstrate the superiority of our TCGL over the state-of-the-art methods on large-scale action recognition and video retrieval benchmarks. The code is publicly available at https://github.com/YangLiu9208/TCGL.**

***Index Terms*—Video Understanding, Self-supervised Learning, Graph Neural Networks.**

I. Introduction

DEEP convolutional neural networks (CNNs) [1] have achieved state-of-the-art performance in many visual recognition tasks. This can be primarily attributed to the learned rich representation from well-trained networks using large-scale image/video datasets (e.g. ImageNet [2], Kinetics [3], SomethingSomething [4]) with strong supervision information [5]. However, annotating such large-scale data is laborious, expensive, and impractical, especially for complex data-based high-level tasks, such as video action understanding and retrieval. To fully leverage the large amount of unlabeled data, self-supervised learning gives a reasonable way to utilize the intrinsic characteristics of unlabeled data to obtain supervisory signals, which has attracted increasing attention.

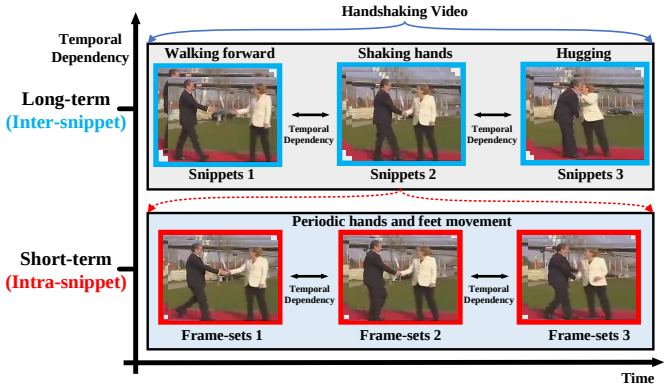

Fig. 1: Illustration of the multi-scale temporal dependencies. The handshaking contains the long-term (inter-snippet) temporal dependencies of walking forward, shaking hands, and hugging, while it also includes the short-term (intra-snippet) temporal dependencies of periodic hands and feet movement.

Different from image data that can be handled by defining proxy tasks (e.g., predicting relative positions of image patches [6], solving jigsaw puzzles [7], inpainting images [8], and predicting the image color channel [9]) for self-supervised learning, video data additionally contain temporal information that can be leveraged to learn the supervisory signals. Recently, a variety of approaches have been proposed such as order verification [10], [11], order prediction [12]–[14], speediness prediction [15], [16]. However, these methods consider the temporal dependency only from a single scale (i.e., short-term or long-term) and ignore the multi-scale temporal dependencies, i.e., they extract either snippet-level or frame-level features via 2D/3D CNNs and neglect to integrate these features to model elaborate multi-scale temporal dependencies.

In this work, we argue that modeling multi-scale temporal dependencies is beneficial for various video classification tasks. Firstly, the recent neuroscience studies [17]–[19] prove that the human visual system can perceive detailed motion information by capturing both long-term and short-term temporal dependencies. This has been inspired by several famous supervised learning methods (e.g., Nonlocal [20], PSANet [21], GloRe [22], and ACNet [23]). Secondly, an action usually consists of several temporal dependencies at both short-term and long-term timescales. Typical actions such as

This work is supported in part by the National Natural Science Foundation of China under Grant No.62002395, in part by the National Natural Science Foundation of Guangdong Province (China) under Grant No. 2021A15150123, and in part by the China Postdoctoral Science Foundation funded project under Grant No.2020M672966. (*Corresponding author: Liang Lin.*)

Yang Liu, Keze Wang, Lingbo Liu, Haoyuan Lan and Liang Lin are with the School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou, China. E-mail: liuy856@mail.sysu.edu.cn; kezewang@gmail.com; liulingb@mail2.sysu.edu.cn; lanhy5@mail2.sysu.edu.cn; linliang@ieee.org.

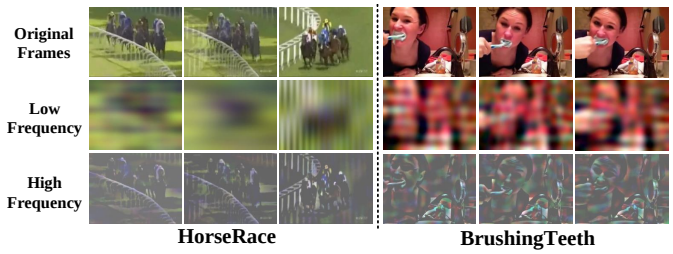

Fig. 2: Illustration of the video frames with different frequencies. The first row is the original frames. The second row is the low frequency (focus on scene representation). And the last row is the high frequency (focus on distinct motion edges).

handshaking and drinking, as well as cycles of repetitive actions such as walking and swimming, often last several seconds and span tens or hundreds of frames. In Figure 1, the video of handshaking contains atomic actions such as walking forward, shaking hands, and hugging, which forms the long-term temporal dependencies (video snippets). Within a snippet, it also includes the short-term temporal dependencies (frame-sets within a snippet) of periodic hands and feet movement. The internal dependency among different snippets and frame-sets contributes to describing the detailed semantic concept for video action analysis. Randomly shuffling the frames or snippets cannot preserve the semantic content of the video. Actually, the short-term temporal dependencies within a video snippet is important especially for videos that contain strict temporal coherence, such as videos in SomethingSomething datasets [4]. Therefore, both short-term (e.g., intra-snippet) and long-term (e.g., inter-snippet) temporal dependencies are essential and should be jointly modeled to learn discriminative temporal representations for unlabeled videos. However, long-term and short-term temporal dependencies have complicated interdependence which are correlated and complementary to each other. Inspired by the convincing performance and high interpretability of graph convolutional networks (GCN) [24]–[28], several works [29]–[32] were proposed to increase the temporal diversity by using GCN in a supervised learning fashion with labeled videos. Unfortunately, due to the lack of principles to explore the multi-scale temporal knowledge of unlabeled videos, it is quite challenging to utilize GCN for self-supervised multi-scale temporal dependencies modeling.

For video frames, their inherent information is conveyed at different frequencies [33]–[35]. As shown in Figure 2, a video frame can be decomposed into a low spatial frequency component that describes the smoothly changing structure (scene representation) and a high spatial frequency that describes the rapidly changing details (motion representation). The low-frequency representation can retain the most of scene information. While in the high frequency, the scene information would be counteracted, and the distinct motion edges would be highlighted. In order to capture both temporal dynamics and scene appearance through different frequencies, we calculate the feature frequency spectrum along the temporal domain based on the discrete cosine transform, and then distill the discriminative spatial-temporal knowledge from videos.

Existing Challenges. Based on the above observations, the following three key challenges should be addressed: (a) how to design a specific feature extraction module that highlights motion information and concurrently preserves both spatial and temporal knowledge from videos? (b) how to model multi-scale temporal dependencies in an explicit way to increase the temporal diversity of large amounts of unlabeled videos? (c) how to build an unified self-supervised learning framework that can learn video understanding model with strong video representative power and generate well to downstream tasks?

Insights. To address aforementioned challenges, this work presents a novel video self-supervised learning approach, named Temporal Contrastive Graph Learning (TCGL). The proposed TCGL aims at learning the multi-scale temporal dependency knowledge within videos by guiding the video snippet order prediction in an adaptive manner. Specifically, a given video is sampled into several fixed-length snippets and then randomly shuffled. For each snippet, all the frames from this snippet are sampled into several fixed-length frame-sets. To address challenge (a), we conduct frequency spectrum analysis using Discrete Cosine Transform (DCT), and propose a novel Spatial-Temporal Knowledge Discovering (STKD) module to highlight motion information and fully discover spatial-temporal information within snippets and frame-sets. Then, we utilize 3D CNN as the backbone network to extract features for these motion-enhanced snippets and frame-sets. To address challenge (b), we propose graph neural network (GNN) structures with prior knowledge about the snippet orders and frame-set orders to explicitly model both inter-snippet and intra-snippet temporal dependencies within videos. The video snippets of a video and their temporal dependencies are used to construct the inter-snippet temporal graph. Similarly, the frame-sets within a video snippet and their temporal characteristics are leveraged to construct the intra-snippet temporal graph. To generate different correlated graph views, we randomly remove edges and mask node features of the intra-snippet graphs or inter-snippet graphs. Then, specific contrastive learning modules are designed to enhance its discriminative capability for temporal representation learning. To address challenge (c), we propose a novel pretext task learning module, named Adaptive Snippet Order Prediction (ASOP) module. The ASOP generates supervisory signals for unlabeled videos by adaptively leveraging relational knowledge among video snippets to predict the actual snippet orders. The main contributions of the paper can be summarized as follows:

- To highlight motion information and discover discriminative spatial-temporal representations from videos, we conduct theoretical analysis in the frequency domain based on discrete cosine transform and propose a novel Spatial-Temporal Knowledge Discovering (STKD) module.
- Integrated with intra-snippet and inter-snippet temporal dependencies, we propose intra-snippet and inter-snippet Temporal Contrastive Graphs (TCG) to increase the temporal diversity among video frames and snippets in a graph contrastive self-supervised learning manner.
- To generate supervisory signals for unlabeled videos, we

propose a novel pretext task learning module, named Adaptive Snippet Order Prediction (ASOP) module, which leverages relational knowledge among video snippets to predict the actual snippet orders by learning the global context representation and recalibrating the channel-wise features adaptively for each video snippet.
- Extensive experiments on three networks and two downstream tasks show that the proposed method achieves state-of-the-art performance and demonstrate the great potential of the learned video representations.

The rest of the paper is organized as follows. We first review related works in Section 2, the details of the proposed method are then 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.

II. Related Work

In this section, we will introduce the recent works on supervised video representation learning, self-supervised video representation learning, and frequency domain learning.

A. Supervised Video Representation Learning

For video representation learning, a large number of supervised learning methods have been received increasing attention. The methods include traditional methods [36]–[44] and deep learning methods [45]–[57]. To model and discover temporal knowledge in videos, two-stream CNNs [45] judged the video image and dense optical flow separately, then directly fused the class scores of these two networks to obtain the classification result. C3D [46] processed videos with a three-dimensional convolution kernel. Temporal Segment Networks (TSN) [49] sampled each video into several segments to model the long-range temporal structure of videos. Temporal Relation Network (TRN) [50] introduced an interpretable network to learn and reason about temporal dependencies between video frames at multiple temporal scales. Temporal Shift Module (TSM) [52] shifted part of the channels along the temporal dimension to facilitate information exchanged among neighboring frames. Although these supervised methods achieve promising performance in modeling temporal dependencies, they require large amounts of labeled videos for training an elaborate model, which is time-consuming and labor-intensive.

B. Self-supervised Video Representation Learning

Although there exists a large amount of videos, it may take a great effort to annotate such massive data. Self-supervised learning gives a feasible way to generate supervisory signals by modeling various pretext tasks with abundant unlabeled data. The learned model from pretext tasks can be directly applied to downstream tasks for feature extraction or fine-tuning. Specific contrastive learning methods have been proposed, such as the NCE [58], MoCo [59], BYOL [60], SimCLR [61]. To better model topologies, contrastive learning methods on graphs [62]–[65] have also been attracted increasing attention. Although the structures of MoCo, SimCLR, BYOL are simple and effective, these methods focus on image recognition without considering the temporal information. Moreover, these methods require large amounts of negative samples or huge batchsize. However, for self-supervised video recognition, videos require much more computational resources than images. Thus, the existing self-supervised image recognition models cannot be directly applied to the video self-supervised learning models due to the limited computational resources. Therefore, we need to elaborately design computationally efficient modules that require less computational resource. Moreover, the videos contain extra temporal information that requires elaborately designed specific modules to explicitly highlight motion information, model temporal dependencies, and generate supervisory signals for unlabeled videos.

For self-supervised video representation learning, how to effectively explore temporal information is important. Many existing works focus on the discovering of temporal information. Shuffle&Learn [10] randomly shuffled video frames and trained a network to distinguish whether these video frames are in the right order or not. Odd-one-out Network [11] proposed to identify unrelated or odd video clips. Order prediction network (OPN) [12] trained networks to predict the correct order of shuffled frames. VCOP [13] used 3D convolutional networks to predict the orders of shuffled video clips. SpeedNet [15] designed a network to detect whether a video is playing at a normal rate or sped up rate. Video-pace [14] utilized a network to identify the right paces of different video clips. In addition to focusing on the temporal dependency, Mas [66] proposed a self-supervised learning method by regressing both motion and appearance statistics along spatial and temporal dimensions. ST-puzzle [5] used space-time cubic puzzles to design pretext task. IIC [67] introduced intra-negative samples by breaking temporal relations in video clips, and used these samples to build an inter-intra contrastive framework. XDC [68] proposed a self-supervised method that leverages unsupervised clustering in audio modality as a supervisory signal for video modality. Though the above works utilize temporal dependency or design specific pretext tasks for video self-supervised learning, the comprehensive temporal diversity and dependency are not fully explored. In our work, we build a novel inter-intra snippet graph structure to model multi-scale temporal dependencies, and produce self-supervision signals about video snippet orders contrastively. Moreover, most of the previous works require stronger models, bigger clip sizes or larger pretrain datasets. While our method can achieve competitive performance with lightweight backbones and smaller clip size.

C. Frequency Domain Learning

Frequency analysis has always been a powerful tool in the signal processing field. Recently, some frequency analysis methods are proposed in deep learning field. Most of frequency-based methods [69], [70] aim to reduce the computing cost and parameters with Fourier Transformation (FT), and therefore improving the network efficiency. Some works [71], [72] introduced frequency analysis for JPEG encoding in the CNNs. DCT is introduced in [73] to reduce the communication bandwidth. Works in [74], [75] proposed dedicated autoencoder-based networks for compression and inference

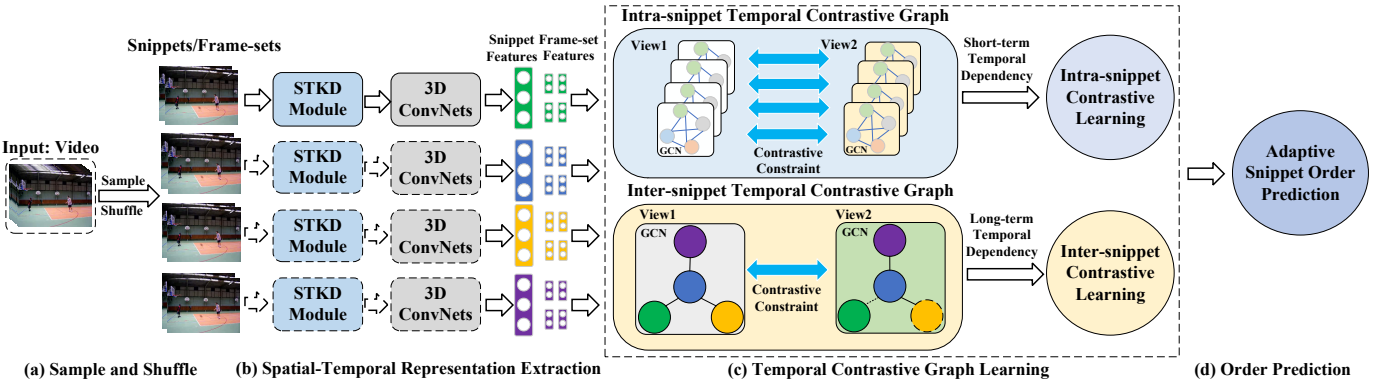

Fig. 3: Overview of the TCGL framework. (a) Sample and Shuffle: sample non-overlapping snippets for each video and randomly shuffle their orders. And for each snippets, all the frames from this snippet are sampled into several fixed-length frame-sets. (b) Spatial-Temporal Representation Extraction: discover the discriminative spatial-temporal representation of video snippets and frame-sets by the STKD module. Then, use the 3D CNNs to extract the features for all the motion-enchanced snippets and frame-sets. (c) Temporal Contrastive Graph Learning: intra-snippet and inter-snippet temporal contrastive graphs are constructed with the prior knowledge about the frame-set orders and snippet orders, see Figure 5 for more details. (d) Order Prediction: the learned snippet features from the temporal contrastive graph are adaptively forwarded through an adaptive snippet order prediction module to output the probability distribution over the possible orders.

tasks. FcaNet [76] generalized the pre-processing of channel attention mechanism in the frequency domain. Wang et al., [77] applied the analysis of the connection between the distribution of frequency components in the input dataset to conduct the explanation of the CNN. Since the convolution operation in the spatial domain has been proven to be equivalent to the multiplication in the frequency domain [78], [79] performed video knowledge distillation in the frequency domain for action recognition. Frequency Filtering Embedding (FFE) [80] used graph Fourier transform and frequency filtering as a graph Fourier domain operator for graph feature extraction. Our method is different from the prior works in two aspects. First, we discover discriminative spatial-temporal knowledge in the frequency domain for video self-supervised learning. Second, we conduct strict theoretical analysis to validate that the discriminative spatial-temporal representation essentially highlights motion information in the high frequency domain.

III. Temporal Contrastive Graph Learning

In this section, we first give a brief overview of the proposed method, then clarify each part of the method in detail. We present the overall framework in Figure 3, which mainly consists of four stages. (1) In sample and shuffle, for each video, several snippets are uniformly sampled and shuffled. For each snippet, all of its frames are sampled into several fixed-length frame-sets. (2) In feature extraction, we discover motion-enhanced spatial-temporal representation of video snippets and frame-sets by the Spatial-Temporal Knowledge Discovering (STKD) module. Then, 3D CNNs are utilized to extract spatial-temporal features for these motion-enhanced snippets and frame-sets, and all 3D CNNs share the same weights. (3) In temporal contrastive learning, we build two kinds of temporal contrastive graph structures (intra-snippet graph and inter-snippet graph) with the prior knowledge about the frame-set orders and snippet orders. To generate different correlated graph views for specific graphs, we randomly remove edges and mask node features of the intra-snippet graphs or inter-snippet graphs. Then, we design specific contrastive losses for both the intra-snippet and inter-snippet graphs to enhance the discriminative capability for discriminative temporal representation learning. (4) In the order prediction, the learned snippet features from the temporal contrastive graph are adaptively forwarded through an adaptive snippet order prediction module to output the probability distribution over the possible orders.

Given a video, the snippets from this video are composed of frames with the size $c \times l \times h \times w$, where c is the number of channels, l is the number of frames, h and w indicate the height and width of frames. The size of the 3D convolutional kernel is $t \times d \times d$, where t is the temporal length and d is the spatial size. We define an ordered snippet tuples as $\mathbf{S} = \langle s_1, s_2, \cdots, s_n \rangle$, the frame-sets from snippet s_i is denoted as $\mathbf{F}_i = \langle f_1, f_2, \cdots, f_m \rangle$. The subscripts here represent the temporal order. Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote a graph, where $\mathcal{V} = \{v_1, v_2, \cdots, v_N\}$ represents the node set and $\mathcal{E} \in \mathcal{V} \times \mathcal{V}$ represents the edge set. We denote the feature matrix and the adjacency matrix as $\mathbf{X} \in \mathbb{R}^{N \times F}$ and $\mathbf{A} \in \{0, 1\}^{N \times N}$, where $\mathbf{x}_i \in \mathbb{R}^F$ is the feature of v_i, and $\mathbf{A}_{ij} = 1$ if $(v_i, v_j) \in \mathcal{E}$.

A. Sample and Shuffle

In this stage, we randomly sample consecutive frames (snippets) from the video to construct video snippet tuples. If we sample N snippets from a video, there are $N!$ possible snippet orders. Since the snippet order prediction is purely a proxy task of our TCGL framework and our focus is the learning of 3D CNNs, we restrict the number of snippets of a video between 3 to 4 to alleviate the complexity of the order prediction task, inspired by the previous works [7], [13], [81]. The snippets are sampled uniformly from the video with the interval of p frames. After sampling, the snippets are shuffled to form the snippet tuples $\mathbf{S} = \langle s_1, s_2, \cdots, s_n \rangle$. For

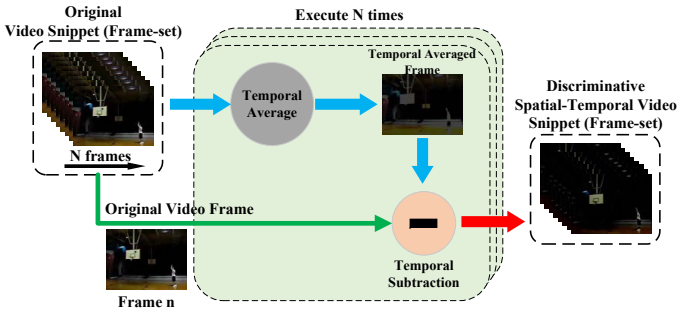

Fig. 4: Illustration of the Spatial-Temporal Knowledge Discovering (STKD) module.

each snippet s_i, all the frames within are uniformly divided into m frame-sets with equal length, then we get the frame-set $\mathbf{F}_i = \langle f_1, f_2, \cdots, f_m \rangle$ for the snippet s_i. For snippet tuples, they contain dynamic information and strict temporal dependency of a video, which is essentially the global temporal structure of the videos. For the frame-sets within a snippet, the frame-level temporal dependency among frames provides us the local temporal structure of the videos. By taking both global and local temporal structures into consideration, we can fully explore discriminative temporal dependencies of videos.

B. Discriminative Spatial-temporal Representation Extraction

As shown in Figure 2, the frequency spectrum computed based on the input video frames can present the motion and scene information with different frequencies. Specifically, high frequency attends to the motion information between neighboring frames, while the low frequency pays attention to the scene representation. Therefore, discriminative spatial-temporal information can be discovered by frequency domain analysis. The overview of the Spatial-Temporal Knowledge Discovering (STKD) module is shown in Figure 4. We denote an input video snippet (frame-set) as $\mathbf{V} \in \mathbb{R}^{C\times L\times H\times W}$, where C is the channel number, L denotes the number of the frames, H and W are the width and height of the video frame, respectively. Then we compute its frequency spectrums along the temporal domain and obtain the output features $\hat{\mathbf{T}} \in \mathbb{R}^{C\times K\times H\times W}$, where K is the number of frequency bands. Typically, the definition of Discrete Cosine Transform (DCT) [82] can be written as:

$$\hat{\mathbf{T}}[k] = \sum_{i=0}^{L-1} \mathbf{V}(i)\cos(\frac{2\pi k i}{L}),\ s.t.\ \ k \in \{0, 1, \cdots, L-1\} \quad (1)$$

Suppose $L = 2$ in Eq (1), we have:

$$\hat{\mathbf{T}}[0] = \mathbf{V}(0) + \mathbf{V}(1) \quad (2)$$

$$\hat{\mathbf{T}}[1] = \mathbf{V}(0) - \mathbf{V}(1) \quad (3)$$

From Eq (2), we can find that $\hat{\mathbf{T}}[0]$ represents the low-frequency information, which is the sum of video features, while $\hat{\mathbf{T}}[1]$ denotes the high-frequency information, which is the difference between neighboring video features. Therefore, the low-frequency representation can retain the most of scene information. While in the high frequency, the scene information would be counteracted, and the distinct motion edges would be highlighted. To further validate this phenomenon, we extend L to 3 and get the following representations:

$$\begin{aligned}\hat{\mathbf{T}}[0] &= \mathbf{V}(0) + \mathbf{V}(1) + \mathbf{V}(2)\\ \hat{\mathbf{T}}[1] &= \mathbf{V}(0) - \frac{1}{2}\mathbf{V}(1) - \frac{1}{2}\mathbf{V}(2)\\ \hat{\mathbf{T}}[2] &= \mathbf{V}(0) - \frac{1}{2}\mathbf{V}(1) - \frac{1}{2}\mathbf{V}(2)\end{aligned} \quad (4)$$

To capture both temporal dynamics and scene appearance through different frequencies, we sum over all the frequency components except for the first one $\hat{\mathbf{T}}[0]$ to avoid the influence of low-frequency information when conduct temporal relationship modeling. Thus, we get the discriminative spatial-temporal representations $\widetilde{\mathbf{T}}_2, \widetilde{\mathbf{T}}_3$ for $L = 2, 3$, respectively:

$$\begin{aligned}\frac{1}{2}\widetilde{\mathbf{T}}_2 &= \mathbf{V}(0) - \frac{1}{2}(\mathbf{V}(0) + \mathbf{V}(1))\\ \frac{1}{3}\widetilde{\mathbf{T}}_3 &= \mathbf{V}(0) - \frac{1}{3}(\mathbf{V}(0) + \mathbf{V}(1) + \mathbf{V}(2))\end{aligned} \quad (5)$$

Thus, when $L = n$, we can obtain:

$$\frac{1}{n}\widetilde{\mathbf{T}}_n = \mathbf{V}(0) - \frac{1}{n}(\mathbf{V}(0) + \mathbf{V}(1) + \cdots + \mathbf{V}(n-1)) \quad (6)$$

Since $\frac{1}{n}$ in the left of the Eq (6) is constant, it can be ignored. Thus, Eq (6) is equivalent to:

$$\widetilde{\mathbf{T}}_n = \mathbf{V}(0) - \frac{1}{n}(\mathbf{V}(0) + \mathbf{V}(1) + \cdots + \mathbf{V}(n-1)) \quad (7)$$

Based on the mathematical induction strategy [83], we can extend Eq (7) to its general form when $L = n + 1$:

$$\widetilde{\mathbf{T}}_{n+1} = \mathbf{V}(0) - \frac{1}{n+1}\sum_{i=0}^{n}\mathbf{V}(i) \quad (8)$$

From Eq (8), we can conclude that the discriminative spatial-temporal representation $\widetilde{\mathbf{T}}_n \in \mathbb{R}^{C\times L\times H\times W}$ for a video snippet is essentially the substraction of the original video snippet $\mathbf{V} \in \mathbb{R}^{C\times L\times H\times W}$ with its temporal average pooling $\overline{\mathbf{V}} = \frac{1}{n+1}\sum_{i=0}^{n}\mathbf{V}(i) \in \mathbb{R}^{C\times L\times H\times W}$ along the temporal axis. The STKD module can extract discriminative temporal representation for video snippets without complex calculation and is flexible enough to be inserted into the existing models in plug-and-play manner. Based on these discriminative spatial-temporal representations for video snippets and frame-sets, we choose C3D [46], R3D-18 [48] and R(2+1)D-18 [48] as feature encoders to further extract spatio-temporal features.

C. Temporal Contrastive Graph Learning Module

Due to the powerful representation ability of graph convolutional network (GCN) [24], [27], [28], [63], [84] in complicated relation modeling, the temporal correlation of intra-/inter-snippet features can be explicitly represented by structured graphs, and their dependencies can be captured by adaptive message propagation through the graphs. In addition, the prior temporal relationship can also determine the edges of graphs more explicitly. Therefore, we use it to explore node interaction within each snippet and frame-set for modeling multi-scale temporal dependencies of videos. After obtaining the feature vectors for snippets and frame-sets, we construct two kinds of temporal contrastive graph

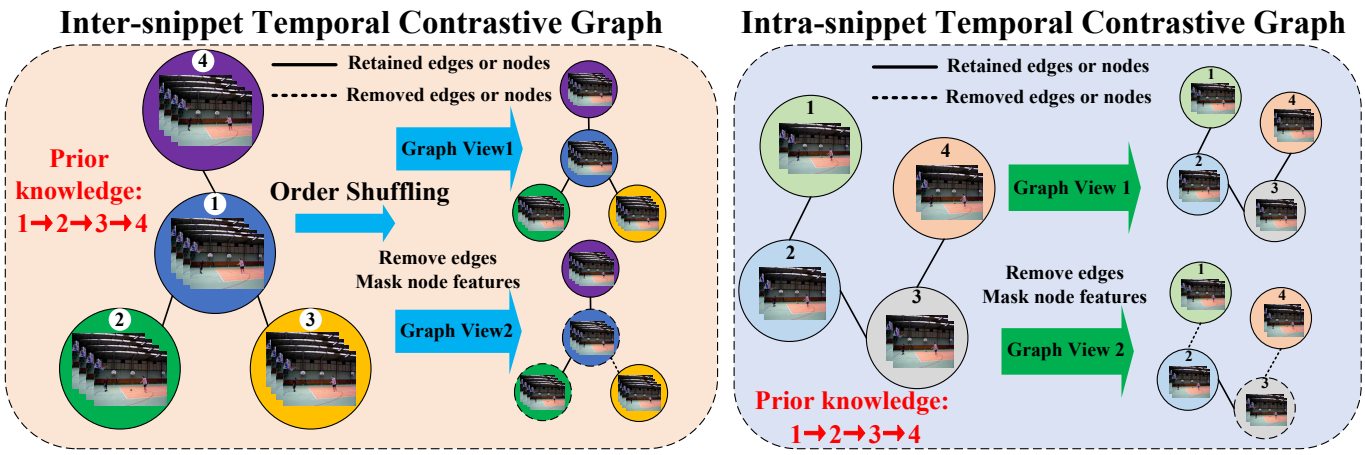

Fig. 5: Illustration of inter-snippet and intra-snippet temporal graphs. Inter-snippet and intra-snippet temporal contrastive graphs are constructed with the prior knowledge about the snippet orders and the frame-set orders of a video.

structures: inter-snippet and intra-snippet temporal contrastive graphs, to increase the temporal diversity of videos, as shown in Figure 3 (c). Assume that a video is shuffled into 4 snippets, and each snippet is further sampling into 4 frame-sets, we can obtain corresponding snippets $\mathbf{S} = \langle s_2, s_4, s_1, s_3 \rangle$ and the frame-sets $\mathbf{F}_i = \langle f_1, f_2, f_3, f_4 \rangle$ for each video snippet $\mathbf{S}_i (i = 1, \cdots, 4)$. Based on snippets features $\mathbf{S}$, frame-sets features $\mathbf{F}_i (i = 1, \cdots, 4)$, and the prior temporal relationship knowledge, we build specific temporal contrastive graphs, as shown in Figure 3 (c).

To build intra-snippet and inter-snippet temporal contrastive graphs, we take advantage of prior knowledge about the temporal relation and the corresponding feature vectors. To fix notation, we denote intra-snippet and inter-snippet graphs as $\mathbf{G}^k_{\text{intra}} = \mathcal{G}(\mathbf{X}^k_{\text{intra}}, \mathbf{A}^k_{\text{intra}})$ and $\mathbf{G}_{\text{inter}} = \mathcal{G}(\mathbf{X}_{\text{inter}}, \mathbf{A}_{\text{inter}})$, respectively, where $k = 1, \cdots, m$, m is the number of frame-sets in a video snippet. As shown in Figure 3, the prior knowledge that the correct order of frames in frame-sets, and the correct order of snippets in snippet tuples are already known because our proxy task is video snippet order prediction. Therefore, we can utilize the prior temporal relationship to determine the edges of graphs. For example, in Figure 5, if we know that snippets (frame-sets) are ranking temporally $1 \rightarrow 2 \rightarrow 3 \rightarrow 4$, we can connect the temporally related nodes and disconnect the temporally unrelated nodes. To jointly discover the dependencies between inter-snippet and intra-snippet graphs and make graph learning consistent, the $\mathbf{G}_{\text{intra}}$ and $\mathbf{G}_{\text{inter}}$ share the same structure but different weight parameters. Here, we take inter-snippet temporal graph $\mathbf{G}_{\text{inter}}$ as an example to clarify our temporal contrastive graph learning method. Both $\mathbf{G}_{\text{inter}}$ and $\mathbf{G}^k_{\text{intra}}$ are undirected graphs.

For $\mathbf{G}_{\text{inter}}$, we randomly remove edges and masking node features to generate two graph views $\widetilde{\mathbf{G}}^1_{\text{inter}}$ and $\widetilde{\mathbf{G}}^2_{\text{inter}}$, and the node embeddings of two generated views are denoted as $\mathbf{U} = \mathcal{G}(\widetilde{\mathbf{X}}^1_{\text{inter}}, \widetilde{\mathbf{A}}^1_{\text{inter}})$ and $\mathbf{V} = \mathcal{G}(\widetilde{\mathbf{X}}^2_{\text{inter}}, \widetilde{\mathbf{A}}^2_{\text{inter}})$. Since different graph views provide different contexts for each node, we corrupt the original graph at both structure and attribute levels to achieve contrastive learning between node embeddings from different views. Therefore, we propose two strategies for generating graph views: removing edges and masking nodes.

The edges in the original graph are randomly removed using a random masking matrix $\widetilde{\mathbf{R}} \in \{0, 1\}^{N \times N}$, where the entry of $\widetilde{\mathbf{R}}$ is drawn from a Bernoulli distribution $\widetilde{\mathbf{R}}_{ij} \sim \mathcal{B}(1 - p_r)$ if $\mathbf{A}_{ij} = 1$ for the original graph, and $\widetilde{\mathbf{R}}_{ij} = 0$ otherwise. Here p_r denotes the probability of each edge being removed. Then, the resulting adjacency matrix can be computed as follows, where $\circ$ is Hadamard product.

$$\widetilde{\mathbf{A}} = \mathbf{A} \circ \widetilde{\mathbf{R}} \tag{9}$$

In addition, a part of node features is masked with zeros using a random vector $\widetilde{\mathbf{m}} \in \{0, 1\}^F$, where each dimension of it is drawn from a Bernoulli distribution $\widetilde{\mathbf{m}}_i \sim \mathcal{B}(1 - p_m), \forall i$. Then, the generated masked features $\widetilde{\mathbf{X}}$ is calculated as follows:

$$\widetilde{\mathbf{X}} = [\mathbf{x}_1 \circ \widetilde{\mathbf{m}}; \mathbf{x}_2 \circ \widetilde{\mathbf{m}}; \cdots; \mathbf{x}_N \circ \widetilde{\mathbf{m}}]^\top \tag{10}$$

where $[\cdot;\cdot]$ is the concatenation operator. We jointly leverage these two strategies to generate graph views.

Different from the NCE loss [58] that only consider inter-view negatives, we take both inter-view and intra-view negative samples into consideration and propose a novel contrastive learning loss that distinguishes embeddings of the same node from these two distinct views from other node embeddings. Given a positive pair, the negative samples come from all other nodes in the two views (inter-view or intra-view). To compute the relationship of embeddings $\mathbf{u}$, $\mathbf{v}$ from two views, we define the relation function $\phi(\mathbf{u}, \mathbf{v}) = \mathcal{P}(g(\mathbf{u}), g(\mathbf{v}))$, where $\mathcal{P}$ is the L2 normalized dot product similarity, and g is a non-linear projection with two-layer multi-layer perception. The pairwise contrastive objective for positive pair $(\mathbf{u}_i, \mathbf{v}_i)$ is defined as:

$$\ell(\mathbf{u}_i, \mathbf{v}_i) = \frac{e^{\frac{\phi(\mathbf{u}_i, \mathbf{v}_i)}{\tau}}}{e^{\frac{\phi(\mathbf{u}_i, \mathbf{v}_i)}{\tau}} + \sum_{k=1}^{N} \mathbb{I}_{[k \neq i]} e^{\frac{\phi(\mathbf{u}_i, \mathbf{v}_k)}{\tau}} + e^{\frac{\phi(\mathbf{u}_i, \mathbf{u}_k)}{\tau}}} \tag{11}$$

where $\mathbb{I}_{[k \neq i]} \in \{0, 1\}$ is an indicator function that equals to 1 if $k \neq i$, and τ is a temperature parameter, which is empirically set to 0.5. The first term in the denominator

represents the positive pairs, the second term represents the inter-view negative pairs, the third term represents the intra-view negative pairs. Since two views are symmetric, the loss for another view is defined similarly for $\ell(\mathbf{v}_i, \mathbf{u}_i)$. The overall contrastive loss for $\mathbf{G}_{\text{inter}}$ is defined as follows:

$$\mathcal{J}_{\text{inter}} = \frac{1}{2N}\sum_{i=1}^{N}[\ell(\mathbf{u}_i, \mathbf{v}_i) + \ell(\mathbf{v}_i, \mathbf{u}_i)] \tag{12}$$

The contrastive loss for intra-snippet graphs $\mathbf{G}^k_{\text{intra}}(k = 1, \cdots, m)$ can be computed similarly as $\mathbf{G}_{\text{inter}}$.

$$\mathcal{J}^k_{\text{intra}} = \frac{1}{2N}\sum_{i=1}^{N}[\ell^k(\mathbf{u}_i, \mathbf{v}_i) + \ell^k(\mathbf{v}_i, \mathbf{u}_i)] \tag{13}$$

Then, the overall temporal contrastive graph loss is defined as follows, where α and β are the weights for intra-snippet graph and inter-snippet graph, respectively.

$$\mathcal{J}_g = \alpha\sum_{k=1}^{m}\mathcal{J}^k_{\text{intra}} + \beta\mathcal{J}_{\text{inter}} \tag{14}$$

D. Adaptive Order Prediction Module

To generate supervisory signals for unlabeled videos, we propose a novel pretext task learning module, named Adaptive Snippet Order Prediction (ASOP). Since the features from different video snippets are correlated, we build an adaptive order prediction module that receives features from different video snippets and learns a global context embedding, then this embedding is used to recalibrate the input features from different snippets, shown in Figure 6. We formulate the order prediction task as a classification task using the learned video snippet features from the temporal contrastive graph as the input and the probability distribution of orders as output.

To fix notation, we assume that a video is shuffled into n snippets, the snippet features of nodes learned from inter-snippet temporal contrastive graph are $\{\mathbf{f}_1, \cdots, \mathbf{f}_n\}$, where $\mathbf{f}_k \in \mathbb{R}^{c_k}(k = 1, \cdots, n)$. To utilize the correlation among these snippets, we concatenate these feature vectors and get joint representations $\mathbf{Z}^k_u$ for each snippet $\mathbf{f}_k$ through a fully-connected layer:

$$\mathbf{Z}^k_u = \mathbf{W}^k_s[\mathbf{f}_1, \cdots, \mathbf{f}_n] + \mathbf{b}^k_s, \quad k = 1, \cdots, n \tag{15}$$

where $[\cdot, \cdot]$ denotes the concatenation operation, $\mathbf{Z}^k_u \in \mathbb{R}^{c_u}$ denotes the joint representation, $\mathbf{W}^k_s$ and $\mathbf{b}^k_s$ are weights and bias of the fully-connected layer. We choose $c_u = \frac{\sum_{k=1}^{n} c_k}{2n}$ to restrict the model capacity and increase its generalization ability. To make use of the global context information aggregated in the joint representations $\mathbf{Z}^k_u$, we predict excitation signal for it via a fully-connected layer:

$$\mathbf{E}^k = \mathbf{W}^k_e\mathbf{Z}^k_u + \mathbf{b}^k_e, \quad k = 1, \cdots, n \tag{16}$$

where $\mathbf{W}^k_e$ and $\mathbf{b}^k_e$ are weights and biases of the fully-connected layer. After obtaining the excitation signal $\mathbf{E}^k \in \mathbb{R}^c$, we use it to recalibrate the input feature $\mathbf{f}_k$ adaptively by a simple gating mechanism:

$$\widetilde{\mathbf{f}}_k = \delta(\mathbf{E}^k) \odot \mathbf{f}_k \tag{17}$$

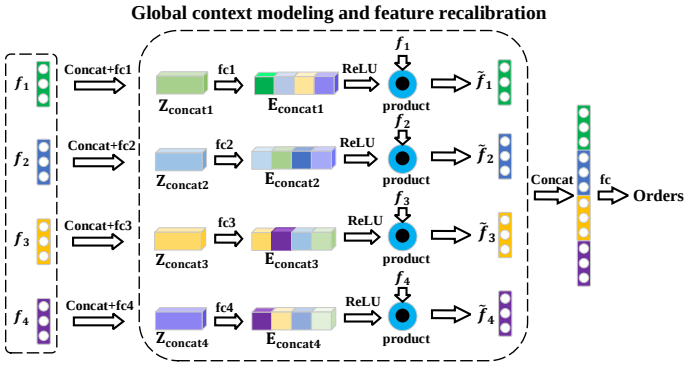

Fig. 6: Adaptive snippet order prediction module with four snippets.

where $\odot$ is channel-wise product operation for each element in the channel dimension, and $\delta(\cdot)$ is the ReLU function. In this way, we can allow the features of one snippet to recalibrate the features of another snippet while concurrently preserving the correlation among different snippets.

Finally, these refined feature vectors $\{\widetilde{\mathbf{f}}_1, \cdots, \widetilde{\mathbf{f}}_n\}$ are concatenated and fed into two-layer perception with soft-max to output the snippet order prediction. To define order classes, we map the snippet orders into different classes. The number of order classes equals to the number of all possible snippet orders. Assume that a video is randomly shuffled into 3 snippets, there are totally 6 (3!) classes. Thus, we can assign class $\{0, 1, 2, 3, 4, 5\}$ to the snippets according to their orders. For example, when the corresponding snippets is $\{s0, s1, s2\}$, its order class is 0. When the corresponding snippets is $\{s2, s1, s0\}$, its order class is 5.

Then, the cross-entropy loss is used to measure the correctness of the prediction:

$$\mathcal{J}_o = -\sum_{i=1}^{C}\mathbf{y}_i\log(\mathbf{p}_i) \tag{18}$$

where $\mathbf{y}_i$ and $\mathbf{p}_i$ represent the probability that the sample belongs to the order class i in ground-truth and prediction, respectively. C denotes the number of all possible orders.

The overall self-supervised learning loss for TCGL is obtained by combing Eq (14) and Eq (18), where λ_g and λ_o control the contribution of $\mathcal{J}_g$ and $\mathcal{J}_o$ respectively as follow:

$$\mathcal{J} = \lambda_g\mathcal{J}_g + \lambda_o\mathcal{J}_o. \tag{19}$$

IV. Experiments

In this section, we first elaborate experimental settings, and then conduct ablation studies to analyze the contribution of key components. Then, the learned 3D CNNs are evaluated on video action recognition and video retrieval tasks with state-of-the-art methods. Finally, we show some visualization results.

A. Experimental Setting

Datasets. We evaluate our method on three action recognition datasets, UCF101 [85], HMDB51 [86], Kinetics-400 [3], Something-Something-V2 [4], and ActivityNet [87]. UCF101

is collected from websites containing 101 action classes with 9.5k videos for training and 3.5k videos for testing. HMDB51 is collected from various sources with 51 action classes and 3.4k videos for training and 1.4k videos for testing. Kinetics-400 (K400) is a large-scale action recognition dataset, which contains 400 action classes and 306k videos. In this work, we use the training split (around 240k videos) as the pre-training dataset. Compared with UCF101, HMDB51 and K400, Something-Something-V2 dataset contains 220,847 videos with 174 classes and focuses more on modeling temporal dependencies. ActivityNet [87] is a large-scale human activity understanding benchmark. The latest released version (v1.3) consists of 19,994 videos from 200 activity categories and is utilized here for evaluation. All the videos in the dataset are divided into 10,024, 4,926, and 5,044 for training, validation, and testing sets, respectively. The labels of testing set are not publicly available and thus the performances on ActivityNet dataset are all reported on validation set.

Network Architecture. For video encoder, C3D, R3D-18 and R(2+1)D-18 are used as backbones, where the kernel size of 3D convolutional layers is set to $3 \times 3 \times 3$. The R3D-18 network is implemented with no repetitions in conv{2-5}_x, which results in 9 convolution layers in total. The C3D network is modified by replacing the two fully connected layers with global spatiotemporal pooling layers. The R(2+1)D-18 network has the same architecture as the R3D-18 network with only 3D kernels decomposed. Dropout layers are applied between fully-connected layers with $p = 0.1$ for R3D-18, and $p = 0.5$ for R3D-50. Our GCN for both inter-snippet and intra-snippet graphs consists of one graph convolutional layer with 512 output channels.

Parameters. Following the settings in [13], [16], we set the snippet length of input video as 16, the interval length is set as 8, the number of snippets per tuple is 3, and the number of frame-sets within each snippet is 4. During training on UCF101 dataset, we follow the same setting as previous works [13], [16], [81] and randomly split 800 videos from the training set as the validation set. When training on K400, Something-Something-V2 and ActivityNet datasets, we use the official validation set. Video frames are resized to 128×171 (256×256) and then randomly cropped to 112×112 (224×224). We set the parameters $\lambda_g = \lambda_o = 1$ to balance the contribution between temporal contrastive graph module and adaptive order prediction module. The weights α and β are both set to 1 according to the ablation study result. To optimize the framework, we use mini-bach stochastic gradient descent with the batchsize 16, the initial learning rate 0.001, the momentum 0.9 and the weight decay 0.0005. The training process lasts for 300 epochs and the learning rate is decreased to 0.0001 after 150 epochs. To make temporal contrastive graphs sensitive to subtle variance between different graph views, the parameters p_r and p_m for generating graph view 1 are empirically set to 0.2 and 0.1, and $p_r = p_m = 0$ for generating graph view 2. And the values of p_r and p_m are the same for both inter-snippet and intra-snippet graphs. The model with the best validation accuracy is saved to the best model. Our method is implemented by PyTorch [88] with eight NVIDIA RTX 3090 GPUs.

Backbone	Snippet Length	Snippets Number	Prediction Accuracy
R3D-18	16	2	53.4
R3D-18	16	3	**83.0**
R3D-18	16	4	61.1

TABLE I: Snippet order prediction accuracy (%) with different number of snippets within each video.

Backbone	Snippet Length	Frame-set Number	Prediction Accuracy
R3D-18	16	1	54.1
R3D-18	16	2	54.1
R3D-18	16	4	**83.0**
R3D-18	16	8	63.1

TABLE II: Snippet order prediction accuracy (%) with various number of frame-sets within each snippet.

Backbone	Snippet Length	Frame-set	GCN Layers	Prediction Accuracy
R3D-18	16	4	1	**83.0**
R3D-18	16	4	2	81.1
R3D-18	16	4	3	54.6

TABLE III: Snippet order prediction accuracy (%) with different number of GCN layers.

Action Recognition and Video retrieval Protocol. To make a fair comparison with other methods, we used the linear probe protocol. Specifically, we directly exploit the backbone learnt by TCGL on UCF101 or Kinetics400 datasets as the pretrained model. Then, we finetune the backbones with the pretrained model using the downstream datasets. Only the linear classifier is randomly initialized and updated, other layers are frozen with the pretrained model.

B. Ablation Study

In this subsection, we conduct ablation studies on the first split of UCF101 with R3D-18 as the backbone, to analyze the contribution of each component of our TCGL and some important hyperparameters. The evaluation metrics are the snippet order prediction accuracy and action recognition accuracy on UCF101 dataset.

The number of snippets. The results of R3D-18 on the snippet order prediction task with different number of snippets are shown in Table I. The prediction accuracy decreases as the number of snippets increases, due to that the difficulty of the prediction task grows when the snippet number increases. Since more snippets makes the model hard to learn, we use 3 snippets per video to make a compromise between the task complexity and prediction accuracy.

The number of frame-sets. Since the snippet length is 16, the number of frame-sets within each snippet can be $1, 2, 4, 8, 16$. When the number is 16, the frame-set only contains static information without temporal information. When the number is 1 or 2, it is hard to model the intra-snippet temporal relationship with too few frame-sets. From Table II, we can observe that more frame-sets within a snippet will make the intra-snippet temporal modeling more difficult, which degrades the order prediction performance. Therefore, we choose 4 frame-sets per snippet in the experiments for short-term temporal modeling.

Backbone	Intra (α)	Inter (β)	ASOP	Prediction	Recognition
R3D-18	0	0	✓	55.6	57.3
R3D-18	0	1	✓	55.6	56.0
R3D-18	1	0	✓	76.6	60.9
R3D-18	0.1	1	✓	54.9	56.7
R3D-18	1	0.1	✓	80.2	66.8
R3D-18	1	1	✓	**83.0**	**76.8**

TABLE IV: Snippet order prediction and action recognition accuracy (%) with different values of α and β.

Backbone	Intra (α)	Inter (β)	ASOP	Prediction	Recognition
R3D-18	1	1	✗	78.4	67.6
R3D-18	1	1	✓	**83.0**	**76.8**

TABLE V: Snippet order prediction and action recognition accuracy (%) with/without ASOP module.

Method	Backbone	Pretrain Dataset	STKD	UCF101	HMDB51
TCGL	C3D	UCF101	✗	69.5	35.1
TCGL	C3D	UCF101	✓	**77.4**	**39.5**
TCGL	C3D	K400	✗	75.2	38.9
TCGL	C3D	K400	✓	**77.9**	**43.0**
TCGL	R3D-18	UCF101	✗	67.6	30.8
TCGL	R3D-18	UCF101	✓	**76.8**	**38.8**
TCGL	R3D-18	K400	✗	76.6	39.5
TCGL	R3D-18	K400	✓	**77.6**	**41.5**
TCGL	R(2+1)D-18	UCF101	✗	74.9	36.2
TCGL	R(2+1)D-18	UCF101	✓	**77.7**	**40.1**
TCGL	R(2+1)D-18	K400	✗	77.6	39.7
TCGL	R(2+1)D-18	K400	✓	**78.5**	**41.4**

TABLE VI: Action recognition accuracy (%) with/without spatial-temporal knowledge discovering (STKD) module.

Backbone	Intra (α)	Inter (β)	ASOP	λ_g	λ_o	Prediction Accuracy
R3D-18	1	1	✓	0	1	77.7
R3D-18	1	1	✓	1	0	21.7
R3D-18	1	1	✓	1	0.1	78.9
R3D-18	1	1	✓	0.1	1	83.0
R3D-18	1	1	✓	1	1	**83.6**

TABLE VII: Snippet order prediction accuracy (%) with different values of λ_g and λ_o.

Backbone	View Generating methods	Prediction	Recognition
R3D-18	Add Random Gaussian Noise	56.1	51.8
R3D-18	Randomly Remove Edges	55.2	53.4
R3D-18	Randomly Remove Nodes	82.9	74.0
R3D-18	Randomly Remove Edges and Nodes	**83.0**	**76.8**

TABLE VIII: Snippet order prediction and action recognition accuracy (%) with different methods of generating views.

The number of GCN layers. From Table III, we can see that more GCN layers will make model converge more difficultly. Since one GCN layer can achieve comparable performance as two GCN layers, we choose one layer GCN.

The intra-snippet and inter-snippet graphs. To analyze the contribution of intra-snippet and inter-snippet temporal contrastive graphs, we set different values to α and β in Eq (14), shown in Table IV. To be noticed, removing intra-snippet graph will degrade the performance significantly even with the inter-snippet graph, which verifies the importance of intra-snippet graphs for modeling short-term temporal dependency. When setting the weight values of intra-snippet graph and inter-snippet graph to 1 and 0.1, the prediction accuracy is 80.2%. After exchanging their weight values, the accuracy drops to 54.9%. This validates that short-term temporal dependency should be attached more importance. In addition, the performance of TCGL drops significantly when removing either or both of the graphs. When $\alpha = \beta = 1$, the prediction accuracy is the best (83.0%). These results validate that both intra-snippet and inter-snippet temporal contrastive graphs are essential for increasing the temporal diversity of features.

The adaptive snippet order prediction. To analyze the contribution of our proposed adaptive snippet order prediction (ASOP) module, we remove this module and merely feed the concatenated features into multi-layer perception with softmax to output the final snippet order prediction. It can be observed in Table V that our TCGL performs better than the TCGL without the ASOP module in both order prediction and action recognition tasks. This verifies that the ASOP module can better utilize relational knowledge among video snippets than simple concatenation.

The spatial-temporal knowledge discovering. To analyze the contribution of our proposed spatial-temporal knowledge discovering (STKD) module, we remove this module and merely feed the raw video frames into the backbones. It can be observed in Table VI that our TCGL performs better than the TCGL without the STKD module across all backbones and datasets. This verifies that the STKD module can discover more discriminative spatial-temporal representations for self-supervised video representation learning, and generalizes well to different backbones and datasets.

The values of λ_g and λ_o. Table VII shows that the performance can drop significantly without either TCG loss or ASOP loss. This validates the importance of both TCG and ASOP modules. The performance is the best when $\lambda_g = \lambda_o = 1$, which shows that we should attach the same importance to graph contrastive learning and snippet order prediction.

The methods of generating views. Actually, how to create graph views is still an open problem. However, randomly masking out snippets neglects the topology among samples. To produce different graph views at both topology and feature levels, we propose an effective strategy that randomly remove edges/nodes. To justify our superiority on modeling both feature and topology levels, we make comparisons with three methods of generating views: (1) adding random Gaussian noise; (2) randomly removing edges on graphs; (3) randomly removing nodes on graphs, as shown in Table VIII. The results in the third row are much better than those in the second row. Actually, the multi-scale temporal dependencies of videos can be explicitly represented by adaptive node message propagation through the graphs. The edges of intra-snippet and inter-snippet temporal contrastive graphs denotes the prior temporal relationship. Therefore, we have the following observations:

(1) Randomly removing the nodes means that the contrastive learning is conduct on sequence of nodes without changing the original temporal graph structure. Therefore, the order

prediction task can still work well for these corrupted nodes.

(2) Since the edges of intra-snippet and inter-snippet temporal contrastive graphs denotes the prior temporal relationship, randomly removing them may change the original temporal dependencies of snippet nodes and thus affect the order prediction accuracy.

(3) However, only randomly masking out snippet nodes neglects the topology among samples. Therefore, we need to produce different graph views at both topology and feature levels. In our TCGL, the proposed hybrid graph contrastive learning strategy can fully utilize the complementary characteristics of feature-level and topology-level graph contrastive learning modules, and make them work collaboratively. To be noticed, the fourth row achieves the best performance. This validates that the feature-level and topology-level graph contrastive learning are complementary and can work collaboratively. Results in Table VIII justify our superiority on modeling both feature and topology levels.

C. Action Recognition

1) Performance on UCF101 and HMDB51: To verify the effectiveness of our TCGL in action recognition, we initialize the backbones with the model pretrained on the first split of UCF101 or the whole K400 training-set, and fine-tune on UCF101 and HMDB51, the fine-tuning stops after 150 epochs. The features extracted by the backbones are fed into fully-connected layers to obtain the prediction. For testing, we sample 10 clips for each video to generate clip predictions, and then average these predictions to obtain the final prediction results. The average classification accuracy over three splits is reported and compared with other self-supervised methods in Table IX, where the backbones, pretrained datasets, and input clip size are illustrated as well. The "Random" means the model is randomly initialized without pre-training.

From Table IX, we can have the following observations: (1) With the same evaluation metric, our TCGL performs favorably against existing approaches under the C3D, R3D-18 and R(2+1)D-18 backbones on both UCF101 and HMDB51 datasets. (2) Compared with random initialization, our TCGL achieves significant improvement on both UCF101 and HMDB51 datasets, which demonstrates the great potential of our TCGL in self-supervised video representation learning. (3) After pre-trained with the C3D backbone, we outperform the current best-performing methods RTT [96] on both UCF101 and HMDB51 datasets. Although the RTT performs slightly better than our TCGL under the R3D-18 and R(2+1)D-18 backbones, our TCGL can achieve more stable performance on UCF101 across three backbones (MAP across three backbones: 78.0%) than the RTT (MAP across three backbones: 76.9%). This validates our good generalization ability for different backbones. In addition, we consistently outperform the other state-of-the-art methods for nearly all evaluation metrics. This validates the scalability and effectiveness of the TCGL. (4) When pre-trained on UCF101 dataset, we achieve better accuracies than some K400 pre-trained methods (Mas [66], ST-puzzle [5] and V-pace [14]). This verifies that modeling multi-scale temporal dependencies can fully discover

Method	Backbone	Pretrain	Clip Size	UCF101	HMDB51
Object Patch [89]	AlexNet	UCF101	227×227	42.7	15.6
Shuffle [10]	CaffeNet	UCF101	224×224	50.9	19.8
OPN [12]	VGG	UCF101	80×80	56.3	22.1
Deep RL [90]	CaffeNet	UCF101	227×227	58.6	25.0
MoCo [59]	I3D	UCF101	16×224×224	70.4	36.3
MemDPC [91]	R2D3D-34	K400	40×224×224	78.1	41.2
SpeedNet [15]	S3D	K400	32×224×224	81.1	48.8
XDC [68]	R(2+1)D-50	K400	32×224×224	86.8	52.6
SeCo [92]	R-50	K400	224 × 224	88.2	55.5
CORP [93]	I3D	K400	32×224×224	90.2	58.7
ELo [94]	R(2+1)D-50	Youtube-8M	N/A	84.2	53.7
Random (Baseline)	C3D	-	16×112×112	61.8	24.7
Mas [66]	C3D	UCF101	16×112×112	58.8	32.6
MoCo [59]	C3D	UCF101	16×112×112	60.5	27.2
VCOP [13]	C3D	UCF101	16×112×112	65.6	28.4
COP [81]	C3D	UCF101	16×112×112	66.9	31.8
PRP [16]	C3D	UCF101	16×112×112	69.1	34.5
STS [95]	C3D	UCF101	16×112×112	69.3	34.2
RTT [96]	C3D	UCF101	16×112×112	68.3	38.4
TCGL (Ours)	C3D	UCF101	16×112×112	<u>77.4</u>	<u>39.5</u>
TCGL (Ours)	C3D	UCF101	16×224×224	**79.5**	**44.4**
ST-puzzle [5]	C3D	K400	16×112×112	60.6	28.3
Mas [66]	C3D	K400	16×112×112	61.2	33.4
CVRL [97]	C3D	K400	32×112×112	69.9	39.6
RTT [96]	C3D	K400	16×112×112	69.9	39.6
STS [95]	C3D	K400	16×112×112	71.8	37.8
RSPNet [98]	C3D	K400	16×112×112	76.7	<u>44.6</u>
TCGL (Ours)	C3D	K400	16×112×112	<u>77.9</u>	43.0
TCGL (Ours)	C3D	K400	16×224×224	**81.0**	**49.2**
Random (Baseline)	R3D-18	-	16×112×112	54.5	23.4
VCOP [13]	R3D-18	UCF101	16×112×112	64.9	29.5
COP [81]	R3D-18	UCF101	16×112×112	66.0	28.0
TCP [99]	R3D-18	UCF101	10×224×224	64.8	34.7
PRP [16]	R3D-18	UCF101	16×112×112	66.5	29.7
STS [95]	R3D-18	UCF101	16×112×112	67.2	32.7
IIC [67]	R3D-18	UCF101	16×112×112	74.4	38.3
RTT [96]	R3D-18	UCF101	16×112×112	77.3	**47.5**
TCGL (Ours)	R3D-18	UCF101	16×112×112	76.8	38.8
TCGL (Ours)	R3D-18	UCF101	16×224×224	<u>78.5</u>	39.4
TCGL (Ours)	R3D-50	UCF101	16×112×112	77.9	39.5
TCGL (Ours)	R3D-50	UCF101	16×224×224	**79.7**	<u>40.4</u>
ST-puzzle [5]	R3D-18	K400	16×112×112	65.8	33.7
DPC [100]	R3D-18	K400	16×128×128	68.2	34.5
DPC [100]	R3D-18	K400	16×224×224	75.7	35.7
TCP [99]	R3D-18	K400	10×224×224	70.5	41.1
RSPNet [98]	R3D-18	K400	16×112×112	74.3	41.8
VideoMoCo [101]	R3D-18	K400	32×112×112	74.1	43.6
RTT [96]	R3D-18	K400	16×112×112	79.3	**49.8**
TCGL (Ours)	R3D-18	K400	16×112×112	77.6	41.5
TCGL (Ours)	R3D-18	K400	16×224×224	<u>80.3</u>	42.8
TCGL (Ours)	R3D-50	K400	16×112×112	78.8	42.0
TCGL (Ours)	R3D-50	K400	16×224×224	**81.5**	<u>43.7</u>
Random (Baseline)	R(2+1)D-18	-	16×112×112	55.8	22.0
VCP [102]	R(2+1)D-18	UCF101	16×112×112	66.3	32.2
VCOP [13]	R(2+1)D-18	UCF101	16×112×112	72.4	30.9
STS [95]	R(2+1)D-18	UCF101	16×112×112	73.6	34.1
COP [81]	R(2+1)D-18	UCF101	16×112×112	74.5	34.8
PRP [16]	R(2+1)D-18	UCF101	16×112×112	72.1	35.0
V-pace [14]	R(2+1)D-18	UCF101	16×112×112	75.9	35.9
PSP [103]	R(2+1)D-18	UCF101	16×112×112	74.8	36.8
TCGL (Ours)	R(2+1)D-18	UCF101	16×112×112	<u>77.7</u>	<u>40.1</u>
TCGL (Ours)	R(2+1)D-18	UCF101	16×224×224	**79.1**	**47.5**
V-pace [14]	R(2+1)D-18	K400	16×112×112	77.1	36.6
STS [95]	R(2+1)D-18	K400	16×112×112	77.8	40.7
VideoMoCo [101]	R(2+1)D-18	K400	32×112×112	78.7	<u>49.2</u>
RSPNet [98]	R(2+1)D-18	K400	16×112×112	81.1	44.6
RTT [96]	R(2+1)D-18	K400	16×112×112	**81.6**	46.4
TCGL (Ours)	R(2+1)D-18	K400	16×112×112	78.5	41.4
TCGL (Ours)	R(2+1)D-18	K400	16×224×224	<u>81.2</u>	**50.1**

TABLE IX: Comparison with the state-of-the-art self-supervised learning methods on UCF101 and HMDB51. The best and the second-best performance are marked in bold and underline styles, respectively.

Method	C3D	R3D-18
W/O pre-training	45.8	42.1
Fully supervised	47.0	43.7
RSPNet [98]	47.8	44.0
TCGL (Ours)	**48.5**	**44.6**

TABLE X: Top-1 action recognition performance on Something-Something-V2 validation set.

Method	Backbone	Pretrain	Batchsize	K400	ActivityNet
MoCo-ImageNet	R-50	ImageNet	512	51.3	66.1
ImageNet Pre-training	R-50	ImageNet	512	52.3	67.1
SeCo-Inter [92]	R-50	ImageNet+K400	512	58.9	66.6
SeCo-Inter+Intra [92]	R-50	ImageNet+K400	512	60.7	68.3
SeCo [92]	R-50	ImageNet+K400	512	61.9	**68.5**
CORP$_m$ [93]	R3D-50	K400	64	59.1	-
CORP$_f$ [93]	R3D-50	K400	512	**66.3**	-
TCGL (Ours)	R3D-50	K400	16	57.6	61.8

TABLE XI: Top-1 linear evaluation results of the proposed TCGL on the Kinetics-400 and ActivityNet datasets.

temporal knowledge of limited unlabeled videos. (5) Although the recent published works SpeedNet [15], XDC [68] and ELo [94] perform better than our TCGL, they require stronger backbones (e.g. S3D, R(2+1)D-50, R2D3D-34), bigger clip size (e.g. 32×224×224), larger pretrain dataset (e.g. Youtube-8M) and bigger batchsize (e.g. 512). While our TCGL can achieve competitive performance with lightweight backbones (C3D, R3D-18, R(2+1)D-18), smaller clip size (16×112×112) and batchsize (16), which is more computationally efficient and require less computational resource. (6) With lightweight backbone and smaller clip size, we achieve comparable accuracy with the MemDPC [91]. (7) When adopting bigger clip size (16×224×224), our TCGL can achieve significant performance improvement. For example, with R(2+1)D-18 pretrained on K400, we achieve 81.2% and 50.1% on UCF101 and HMDB51, respectively. (8) The TCGL with R3D-50 backbone can obtain nearly 1% and 0.5% performance gains than that of the R3D-18 backbone on UCF101 and HMDB51 datasets, respectively. This validates that the R3D-50 backbone is more powerful than the R3D-18 for spatial-temporal representation learning. These observations validate the advantages of our TCGL in learning discriminative spatial-temporal representations for self-supervised video action recognition.

2) Performance on Something-Something-V2: To verify the effectiveness of our TCGL in action recognition scenario which requires strict temporal modeling, we initialize the backbones with the model pretrained the whole K400 training-set, and then fine-tune on Something-Something-V2. The fine-tuning stops after 150 epochs. The features extracted by the backbones are fed into fully-connected layers to obtain the prediction. For the w/o pretraining methods, the models are randomly initialized. For the fully supervised methods, the C3D and R3D-18 are fully trained on K400 dataset, which is a large-scale dataset with manually annotated action labels, and thus the supervised pre-trained models are strong baselines for our unsupervised pre-trained TCGL. In Table X, despite not using manual annotations, TCGL increases the accuracy compared with the random initialized model. Surprisingly, with C3D and R3D-18, our TCGL even outperforms the supervised pretrained model, increasing from (47.0%, 43.7%) to (48.5%, 44.6%), respectively. In addition, we also perform better than the self-supervised method RSPNet [98] under the same setting across C3D and R3D-18 backbones. These results show that our TCGL can learn discriminative spatial-temporal representations on action recognition dataset that requires strict temporal modeling, which validates the great potential of our TCGL in self-supervised video understanding.

3) Performance on Kinetics400 and ActivityNet datasets: To verify the effectiveness of our TCGL on downstream tasks of action recognition and untrimmed activity recognition, we follow the settings of [92] and pretrain the R3D-50 backbone by TCGL on Kinetics400, and then finetune the pretrained R3D-50 backbone with the downstream datasets Kinetics400 and ActivityNet, respectively. All layers except the last linear layer are frozen with the pre-training backbone. The finetuned model is exploited as the feature extractor to verify the frozen representation via linear classification. For each video in Kinetics400 and ActivityNet, we uniformly sample 30 and 50 frames, respectively, resize each frame with short edge of 256, and crop the resized version to 224×224 by using center crop. The linear classifier is finally trained on the training videos of Kinetics400 or ActivityNet and evaluated on each validation set. We adopt the top-1 accuracy as the performance metric, as shown in Table XI.

On Kinetics400 (K400) dataset, our TCGL can achieve performance improvement over MoCo-ImageNet and ImageNet Pre-training. This validates that our TCGL can learn discriminative spatial-temporal representations on Kinetics400 dataset. Since large batch sizes can result in better performances for contrastive learning [92], the CORP$_f$ achieves the best accuracy by using big batchsize (512) and pretraining the model for 800 epochs. The SeCo [92] also performs better than our TCGL. This is because that the SeCo also use big batchsize (512) and pretrains the backbone initialized with MoCo on ImageNet. The weights initialized from MoCO on ImageNet can facilitate the pretraining of the SeCo model and thus improve its discriminative power. In addition, the CORP and SeCo need extra data augmentations like random cropping with random scales, color-jitter, random grayscale, blur, and mirror. Different from the CORP and SeCo, our TCGL requires no extra data augmentations and pretrains the backbones from scratch (with random initialized weights), smaller batchsize (16), and fewer pretraining epochs (300). Nonetheless, our TCGL can still achieve comparable performance with CORP$_m$ with similar settings.

On Activity dataset, the SeCo achieves the best performance and performs better than our TCGL due to the fact that the SeCo pretrains the backbone initialized with MoCo on ImageNet and use big batchsize and extra data augmentations. Although pretraining from scratch with small batchsize and no data augmentations, our TCGL can achieve the accuracy of 61.8% on untrimmed activity recognition dataset ActivityNet, which is more computationally efficient and require less computational resource. These observations validate the advantages of our TCGL in learning discriminative spatial-

Fig. 7: Video retrieval results with TCGL representations on UCF101 and HMDB51 datasets. The first column contains query clips from the test split, and the remaining columns indicate top-2 nearest clips retrieved by different trained models from the training split. The class of each video is displayed in bottom. The class name in red color denotes the correct retrieval class, while the class name in green color denotes the wrong retrieval class.

Method	Backbone	Pretrain	Top1	Top5	Top10	Top20	Top50
Jigsaw [7]	AlexNet	UCF101	19.7	28.5	33.5	40.0	49.4
OPN [12]	VGG	UCF101	19.9	28.7	34.0	40.6	51.6
Deep RL [90]	CaffeNet	UCF101	25.7	36.2	42.2	49.2	59.5
SpeedNet [15]	S3D	K400	13.0	28.1	37.5	49.5	65.0
Random	C3D	UCF101	16.7	27.5	33.7	41.4	53.0
VCOP [13]	C3D	UCF101	12.5	29.0	39.0	50.6	66.9
PRP [16]	C3D	UCF101	23.2	38.1	46.0	55.7	68.4
V-pace [14]	C3D	UCF101	20.0	37.4	46.9	58.5	73.1
TCGL (Ours)	C3D	UCF101	**22.4**	**41.3**	**51.0**	**61.4**	**75.0**
TCGL (Ours)	C3D	K400	**23.6**	**41.5**	**52.0**	**61.8**	**75.2**
Random	R3D-18	UCF101	9.9	18.9	26.0	35.5	51.9
VCOP [13]	R3D-18	UCF101	14.1	30.3	40.4	51.1	66.5
PRP [16]	R3D-18	UCF101	22.8	38.5	46.7	55.2	69.1
V-pace [14]	R3D-18	UCF101	19.9	36.2	46.1	55.6	69.2
MemDPC [91]	R3D-18	UCF101	20.2	40.4	52.4	64.7	-
TCGL (Ours)	R3D-18	UCF101	**23.4**	**42.2**	**51.9**	**62.0**	**74.8**
TCGL (Ours)	R3D-18	K400	**23.9**	**43.0**	**53.0**	**62.9**	**75.7**
Random	R(2+1)D-18	UCF101	10.6	20.7	27.4	37.4	53.1
VCOP [13]	R(2+1)D-18	UCF101	10.7	25.9	35.4	47.3	63.9
PRP [16]	R(2+1)D-18	UCF101	20.3	34.0	41.9	51.7	64.2
V-pace [14]	R(2+1)D-18	UCF101	17.9	34.3	44.6	55.5	72.0
TCGL (Ours)	R(2+1)D-18	UCF101	**21.5**	**39.3**	**49.3**	**59.5**	**72.7**
TCGL (Ours)	R(2+1)D-18	K400	**21.9**	**40.2**	**49.6**	**59.7**	**73.1**

TABLE XII: Video retrieval result (%) on UCF101.

Method	Backbone	Pretrain	Top1	Top5	Top10	Top20	Top50
Random	C3D	UCF101	7.4	20.5	31.9	44.5	66.3
VCOP [13]	C3D	UCF101	7.4	22.6	34.4	48.5	70.1
PRP [16]	C3D	UCF101	10.5	27.2	40.4	56.2	75.9
V-pace [14]	C3D	UCF101	8.0	25.2	37.8	54.4	77.5
MoCo+BE [104]	C3D	UCF101	10.2	27.6	40.5	56.2	76.6
TCGL (Ours)	C3D	UCF101	**10.7**	**28.6**	**41.1**	**57.9**	**77.7**
TCGL (Ours)	C3D	K400	**12.3**	**30.4**	**42.9**	**59.1**	**79.2**
Random	R3D-18	UCF101	6.7	18.3	28.3	43.1	67.9
VCOP [13]	R3D-18	UCF101	7.6	22.9	34.4	48.8	68.9
PRP [16]	R3D-18	UCF101	8.2	25.8	38.5	53.3	75.9
V-pace [14]	R3D-18	UCF101	8.2	24.2	37.3	53.3	74.5
MemDPC [91]	R3D-18	UCF101	7.7	25.7	40.6	57.7	-
TCGL (Ours)	R3D-18	UCF101	**11.7**	**28.9**	**40.5**	**55.4**	**76.8**
TCGL (Ours)	R3D-18	K400	**12.1**	**30.6**	**43.8**	**58.1**	**78.0**
Random	R(2+1)D-18	UCF101	4.5	14.8	23.4	38.9	63.0
VCOP [13]	R(2+1)D-18	UCF101	5.7	19.5	30.7	45.8	67.0
PRP [16]	R(2+1)D-18	UCF101	8.2	25.3	36.2	51.0	73.0
V-pace [14]	R(2+1)D-18	UCF101	10.1	24.6	37.6	54.4	77.1
TCGL (Ours)	R(2+1)D-18	UCF101	**10.5**	**27.6**	**39.7**	**55.6**	**76.4**
TCGL (Ours)	R(2+1)D-18	K400	**11.1**	**30.4**	**43.0**	**56.5**	**77.4**

TABLE XIII: Video retrieval result (%) on HMDB51.

temporal representations on downstream tasks of action recognition and untrimmed activity recognition.

D. Video Retrieval

To further verify the effectiveness of our TCGL in video retrieval, we test our TCGL on the nearest-neighbor video retrieval. Since the video retrieval task is conducted with features extracted by the backbone network without fine-tuning, its performance largely relies upon the representative capacity of self-supervised models. The experiment is conducted on the first split of UCF101 or the whole training-set of K400, following the protocol in [13], [16]. In video retrieval, we extract video features from the backbone pre-trained by TCGL. Each video in the testing set is used to query k nearest videos from the training set using the cosine distance. When the class of a test video appears in the classes of k nearest training videos, it is considered as the correct predicted video. We show top-1, top-5, top-10, top-20, and top-50 retrieval accuracies on UCF101 and HMDB51 datasets, and compare our method with other self-supervised methods, as shown in Table XII and XIII. For all backbones, our TCGL outperforms the state-of-the-art methods by substantial margins for nearly all metrics. To be noticed, although the SpeedNet [15] is trained on more powerful backbone S3D, our TCGL performs

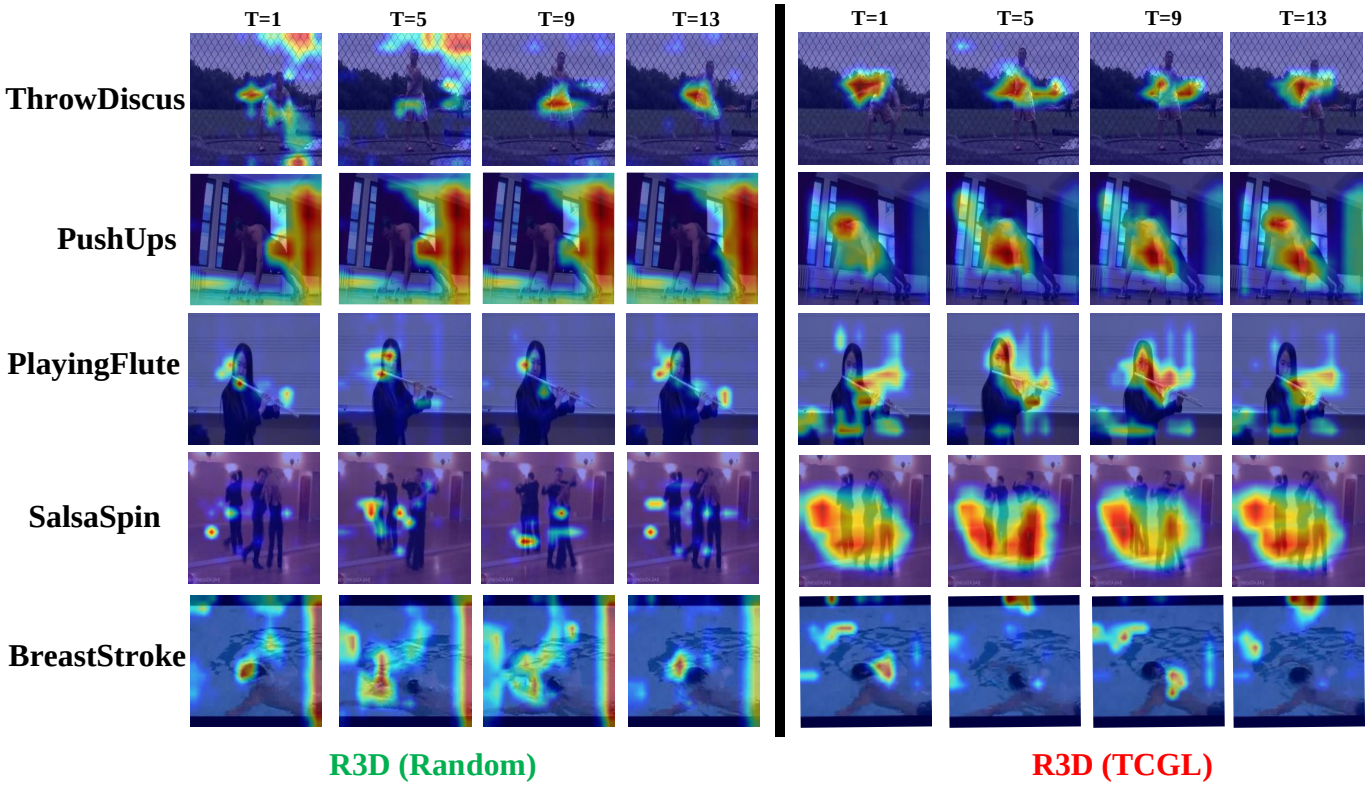

Fig. 8: Visualization of a sequence of activation maps on UCF101 dataset with R3D-18 as the backbone.

better. These observations validate our proposed TCGL can learn discriminative representations for video retrieval task.

E. Visualization Analysis

To have an intuitive understanding of the video retrieval results, Figure 7 visualizes a query video snippet and its top-2 nearest neighbors from the UCF101 dataset. The leftmost columns are videos used for the query, and the remaining columns show top-2 retrieved videos by different feature extractors. In most cases, these retrieved videos portray a strong semantic correlation with the query videos, even when performing complex interactive actions in the presence of significant camera motion, background clutter and occlusion. To be noticed, in some challenging cases, our TCGL fails to retrieve true videos due to the high appearance and semantic similarity. For basketball video, the R(2+1)D network finds volleyball spiking video which is also sports and contains balls. For playing violin video, the C3D network retrieves playing cello video and the R(2+1)D network retrieves playing guitar video, which are also musical activities and contain musical instruments with similar appearance. For taichi video, due to the existence of similar atomic human actions such as crouch, punch, and kick, the C3D, R3D, R(2+1)D networks retrieve punch, boxingspeedbag and yoyo videos, respectively. For ridebike video, due to the similar background color and action movement style, the C3D, R3D, R(2+1)D networks retrieve ridehorse, handshake and run videos, respectively.

To have a better understanding of what TCGL learns, we follow the Class Activation Map [105] to visualize the spatio-temporal regions, as shown in Figure 8. These examples exhibit a strong correlation between highly activated regions and the dominant movement in the scene. This validates that our TCGL can effectively capture the dominant movement in the videos by learning discriminative temporal representations for videos. To be noticed, in some challenging videos like salsaspin and breaststroke, our TCGL fails to capture true moving regions. In salsaspin videos, our TCGL takes the moving areas in the mirror as the activated regions. In breaststroke videos, our TCGL fails to focus on dominant movement of swimming people due to the disturbance of the motion of the water.

V. Conclusion

In this paper, we proposed a novel Temporal Contrastive Graph Learning (TCGL) approach for self-supervised video representation learning. We fully discover discriminative spatial-temporal representation by strict theoretical analysis in the frequency domain based on discrete cosine transform. Integrated with intra-snippet and inter-snippet temporal dependencies, we introduced intra-snippet and inter-snippet temporal contrastive graphs to increase the temporal diversity among video frames and snippets in a graph contrastive self-supervised learning manner. To learn the global context representation and recalibrate the channel-wise features adaptively for each video snippet, we proposed an adaptive video snippet order prediction module, which employs the relational knowledge among video snippets to predict orders. With inter-intra snippet graph contrastive learning strategy and adaptive video snippet order prediction task, the temporal diversity and multi-scale temporal dependency can be well discovered. The proposed TCGL is applied to video action recognition and

video retrieval tasks with three kinds of 3D CNNs. Extensive experiments demonstrate the superiority of our TCGL over the state-of-the-art methods on large-scale benchmarks. Future direction will be the evaluation of our method with more powerful backbones, larger pretrained datasets, and more downstream tasks.

REFERENCES

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

[2] 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.

[3] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev *et al.*, "The kinetics human action video dataset," *arXiv preprint arXiv:1705.06950*, 2017.

[4] R. Goyal, S. E. Kahou, V. Michalski, J. Materzynska, S. Westphal, H. Kim, V. Haenel, I. Fruend, P. Yianilos, M. Mueller-Freitag *et al.*, "The" something something" video database for learning and evaluating visual common sense." in *ICCV*, vol. 1, no. 4, 2017, p. 5.

[5] 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.

[6] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1422–1430.

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

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

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

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

[11] 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.

[12] 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.

[13] 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.

[14] J. Wang, J. Jiao, and Y.-H. Liu, "Self-supervised video representation learning by pace prediction," in *ECCV*, 2020, pp. 504–521.

[15] 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.

[16] 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.

[17] M. Livingstone and D. Hubel, "Segregation of form, color, movement, and depth: anatomy, physiology, and perception," *Science*, vol. 240, no. 4853, pp. 740–749, 1988.

[18] D. C. Van Essen and J. L. Gallant, "Neural mechanisms of form and motion processing in the primate visual system," *Neuron*, vol. 13, no. 1, pp. 1–10, 1994.

[19] J. Hans and J. R. Cruysberg, "The visual system," in *Clinical Neuroanatomy*. Springer, 2020, pp. 409–453.

[20] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7794–7803.

[21] H. Zhao, Y. Zhang, S. Liu, J. Shi, C. C. Loy, D. Lin, and J. Jia, "Psanet: Point-wise spatial attention network for scene parsing," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 267–283.

[22] Y. Chen, M. Rohrbach, Z. Yan, Y. Shuicheng, J. Feng, and Y. Kalantidis, "Graph-based global reasoning networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 433–442.

[23] G. Wang, K. Wang, and L. Lin, "Adaptively connected neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1781–1790.

[24] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[25] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *International Conference on Learning Representations*, 2018.

[26] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: A survey," *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[27] Y. Yang, J. Qiu, M. Song, D. Tao, and X. Wang, "Distilling knowledge from graph convolutional networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 7074–7083.

[28] Y. Yang, Z. Feng, M. Song, and X. Wang, "Factorizable graph convolutional networks," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[29] X. Wang and A. Gupta, "Videos as space-time region graphs," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 399–417.

[30] T. Zhuo, Z. Cheng, P. Zhang, Y. Wong, and M. Kankanhalli, "Explainable video action reasoning via prior knowledge and state transitions," in *Proceedings of the 27th ACM International Conference on Multimedia*, 2019, pp. 521–529.

[31] J. Ji, R. Krishna, L. Fei-Fei, and J. C. Niebles, "Action genome: Actions as compositions of spatio-temporal scene graphs," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 236–10 247.

[32] J. Zhang, F. Shen, X. Xu, and H. T. Shen, "Temporal reasoning graph for activity recognition," *IEEE Transactions on Image Processing*, vol. 29, pp. 5491–5506, 2020.

[33] F. W. Campbell and J. G. Robson, "Application of fourier analysis to the visibility of gratings," *The Journal of physiology*, vol. 197, no. 3, p. 551, 1968.

[34] R. L. De Valois and K. K. De Valois, "Spatial vision," *Annual review of psychology*, vol. 31, no. 1, pp. 309–341, 1980.

[35] Y. Chen, H. Fan, B. Xu, Z. Yan, Y. Kalantidis, M. Rohrbach, S. Yan, and J. Feng, "Drop an octave: Reducing spatial redundancy in convolutional neural networks with octave convolution," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 3435–3444.

[36] I. Laptev, "On space-time interest points," *International journal of computer vision*, vol. 64, no. 2-3, pp. 107–123, 2005.

[37] A. Klaser, M. Marszałek, and C. Schmid, "A spatio-temporal descriptor based on 3d-gradients," in *BMVC 2008-19th British Machine Vision Conference*, 2008, pp. 275–1.

[38] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, "Dense trajectories and motion boundary descriptors for action recognition," *International journal of computer vision*, vol. 103, no. 1, pp. 60–79, 2013.

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

[40] T. V. Nguyen, Z. Song, and S. Yan, "Stap: Spatial-temporal attention-aware pooling for action recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 25, no. 1, pp. 77–86, 2014.

[41] Y. Liu, J. Li, Z. Lu, T. Yang, and Z. Liu, "Combining multiple features for cross-domain face sketch recognition," in *Chinese Conference on Biometric Recognition*, 2016, pp. 139–146.

[42] X. Peng, L. Wang, X. Wang, and Y. Qiao, "Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice," *Computer Vision and Image Understanding*, vol. 150, pp. 109–125, 2016.

[43] Y. Liu, Z. Lu, J. Li, C. Yao, and Y. Deng, "Transferable feature representation for visible-to-infrared cross-dataset human action recognition," *Complexity*, vol. 2018, 2018.

[44] Y. Liu, Z. Lu, J. Li, and T. Yang, "Hierarchically learned view-invariant representations for cross-view action recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 8, pp. 2416–2430, 2018.

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

[46] 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.

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

[48] 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.

[49] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, "Temporal segment networks for action recognition in videos," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 11, pp. 2740–2755, 2018.

[50] B. Zhou, A. Andonian, A. Oliva, and A. Torralba, "Temporal relational reasoning in videos," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 803–818.

[51] Y. Liu, Z. Lu, J. Li, T. Yang, and C. Yao, "Global temporal representation based cnns for infrared action recognition," *IEEE Signal Processing Letters*, vol. 25, no. 6, pp. 848–852, 2018.

[52] J. Lin, C. Gan, and S. Han, "Tsm: Temporal shift module for efficient video understanding," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 7083–7093.

[53] Y. Liu, Z. Lu, J. Li, T. Yang, and C. Yao, "Deep image-to-video adaptation and fusion networks for action recognition," *IEEE Transactions on Image Processing*, vol. 29, pp. 3168–3182, 2020.

[54] C.-G. Huang, H.-Z. Huang, and Y.-F. Li, "A bidirectional lstm prognostics method under multiple operational conditions," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 11, pp. 8792–8802, 2019.

[55] Y. Liu, K. Wang, G. Li, and L. Lin, "Semantics-aware adaptive knowledge distillation for sensor-to-vision action recognition," *IEEE Transactions on Image Processing*, vol. 30, pp. 5573–5588, 2021.

[56] J. Ni, R. Sarbajna, Y. Liu, A. H. Ngu, and Y. Yan, "Cross-modal knowledge distillation for vision-to-sensor action recognition," *arXiv preprint arXiv:2112.01849*, 2021.

[57] C.-G. Huang, H.-Z. Huang, Y.-F. Li, and W. Peng, "A novel deep convolutional neural network-bootstrap integrated method for rul prediction of rolling bearing," *Journal of Manufacturing Systems*, 2021.

[58] M. Gutmann and A. Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 2010, pp. 297–304.

[59] 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.

[60] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. D. Guo, M. G. Azar *et al.*, "Bootstrap your own latent: A new approach to self-supervised learning," *arXiv preprint arXiv:2006.07733*, 2020.

[61] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton, "A simple framework for contrastive learning of visual representations," in *Proceedings of the 37th International Conference on Machine Learning,*, 2020, pp. 1597–1607.

[62] J. Qiu, Q. Chen, Y. Dong, J. Zhang, H. Yang, M. Ding, K. Wang, and J. Tang, "Gcc: Graph contrastive coding for graph neural network pre-training," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 1150–1160.

[63] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, and L. Wang, "Deep graph contrastive representation learning," *arXiv preprint arXiv:2006.04131*, 2020.

[64] H. Hafidi, M. Ghogho, P. Ciblat, and A. Swami, "Graphcl: Contrastive self-supervised learning of graph representations," *arXiv preprint arXiv:2007.08025*, 2020.

[65] Y. You, T. Chen, Y. Sui, T. Chen, Z. Wang, and Y. Shen, "Graph contrastive learning with augmentations," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[66] 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.

[67] 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.

[68] H. Alwassel, D. Mahajan, B. Korbar, L. Torresani, B. Ghanem, and D. Tran, "Self-supervised learning by cross-modal audio-video clustering," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[69] Z. Jia, A. Zlateski, F. Durand, and K. Li, "Optimizing n-dimensional, winograd-based convolution for manycore cpus," in *Proceedings of the 23rd ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, 2018, pp. 109–123.

[70] L. Wang, W. Wu, J. Zhang, H. Liu, G. Bosilca, M. Herlihy, and R. Fonseca, "Fft-based gradient sparsification for the distributed training of deep neural networks," in *Proceedings of the 29th International Symposium on High-Performance Parallel and Distributed Computing*, 2020, pp. 113–124.

[71] L. Gueguen, A. Sergeev, B. Kadlec, R. Liu, and J. Yosinski, "Faster neural networks straight from jpeg," *Advances in Neural Information Processing Systems*, vol. 31, pp. 3933–3944, 2018.

[72] M. Ehrlich and L. S. Davis, "Deep residual learning in the jpeg transform domain," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 3484–3493.

[73] K. Xu, M. Qin, F. Sun, Y. Wang, Y.-K. Chen, and F. Ren, "Learning in the frequency domain," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1740–1749.

[74] R. Torfason, F. Mentzer, E. Agustsson, M. Tschannen, R. Timofte, and L. V. Gool, "Towards image understanding from deep compression without decoding," in *International Conference on Learning representations*, 2018.

[75] C.-Y. Wu, M. Zaheer, H. Hu, R. Manmatha, A. J. Smola, and P. Krähenbühl, "Compressed video action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6026–6035.

[76] Z. Qin, P. Zhang, F. Wu, and X. Li, "Fcanet: Frequency channel attention networks," *arXiv preprint arXiv:2012.11879*, 2020.

[77] Z. Wang, Y. Yang, A. Shrivastava, V. Rawal, and Z. Ding, "Towards frequency-based explanation for robust cnn," *arXiv preprint arXiv:2005.03141*, 2020.

[78] R. N. Bracewell and R. N. Bracewell, *The Fourier transform and its applications*. McGraw-Hill New York, 1986, vol. 31999.

[79] H. Su, J. Su, D. Wang, W. Gan, W. Wu, M. Wang, J. Yan, and Y. Qiao, "Collaborative distillation in the parameter and spectrum domains for video action recognition," *arXiv preprint arXiv:2009.06902*, 2020.

[80] H. Bahonar, A. Mirzaei, S. Sadri, and R. C. Wilson, "Graph embedding using frequency filtering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 2, pp. 473–484, 2021.

[81] J. Xiao, L. Li, D. Xu, C. Long, J. Shao, S. Zhang, S. Pu, and Y. Zhuang, "Explore video clip order with self-supervised and curriculum learning for video applications," *IEEE Transactions on Multimedia*, 2020.

[82] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE transactions on Computers*, vol. 100, no. 1, pp. 90–93, 1974.

[83] J. A. Bather, "Mathematical induction," 1994.

[84] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.

[85] 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.

[86] 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.

[87] 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.

[88] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[89] X. Wang and A. Gupta, "Unsupervised learning of visual representations using videos," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2794–2802.

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

[91] T. Han, W. Xie, and A. Zisserman, "Memory-augmented dense predictive coding for video representation learning," *arXiv preprint arXiv:2008.01065*, 2020.
[92] T. Yao, Y. Zhang, Z. Qiu, Y. Pan, and T. Mei, "Seco: Exploring sequence supervision for unsupervised representation learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 12, 2021, pp. 10 656–10 664.
[93] K. Hu, J. Shao, Y. Liu, B. Raj, M. Savvides, and Z. Shen, "Contrast and order representations for video self-supervised learning," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 7939–7949.
[94] 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.
[95] J. Wang, J. Jiao, L. Bao, S. He, W. Liu, and Y.-H. Liu, "Self-supervised video representation learning by uncovering spatio-temporal statistics," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.
[96] S. Jenni, G. Meishvili, and P. Favaro, "Video representation learning by recognizing temporal transformations," in *European Conference on Computer Vision*, 2020, pp. 425–442.
[97] R. Qian, T. Meng, B. Gong, M.-H. Yang, H. Wang, S. Belongie, and Y. Cui, "Spatiotemporal contrastive video representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 6964–6974.
[98] P. Chen, D. Huang, D. He, X. Long, R. Zeng, S. Wen, M. Tan, and C. Gan, "Rspnet: Relative speed perception for unsupervised video representation learning," in *AAAI Conference on Artificial Intelligence*, vol. 1, 2021.
[99] G. Lorre, J. Rabarisoa, A. Orcesi, S. Ainouz, and S. Canu, "Temporal contrastive pretraining for video action recognition," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020, pp. 662–670.
[100] 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. 1–10.
[101] T. Pan, Y. Song, T. Yang, W. Jiang, and W. Liu, "Videomoco: Contrastive video representation learning with temporally adversarial examples," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 11 205–11 214.
[102] D. Luo, C. Liu, Y. Zhou, D. Yang, C. Ma, Q. Ye, and W. Wang, "Video cloze procedure for self-supervised spatio-temporal learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, 2020, pp. 11 701–11 708.
[103] 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.
[104] J. Wang, Y. Gao, K. Li, Y. Lin, A. J. Ma, H. Cheng, P. Peng, F. Huang, R. Ji, and X. Sun, "Removing the background by adding the background: Towards background robust self-supervised video representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 11 804–11 813.
[105] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2921–2929.

Yang Liu (M'21) is currently an associate research fellow in the School of Computer Science and Engineering, Sun Yat-Sen University, working with Prof. Liang Lin. He was a postdoctoral fellow in the School of Computer Science and Engineering, Sun Yat-Sen University in 2019-2021. He received the Ph.D. degree in telecommunications and information systems from Xidian University, Xi'an, China in 2019, advised by Prof. Zhaoyang Lu. He received the B.S. degree in telecommunications engineering from Chang'an University, Xi'an, China, in 2014. His current research interests include computer vision and machine learning. He has authored and coauthored more than 10 papers in top-tier academic journals and conferences, including IEEE T-IP, IEEE T-CSVT, and IEEE SPL, etc. More information can be found on https://yangliu9208.github.io/home.

Keze Wang is currently an associate professor in the School of Computer Science and Engineering, Sun Yat-Sen University. He received his B.S. degree in software engineering from Sun Yat-Sen University, Guangzhou, China, in 2012. He obtained my Ph.D. degree with honors from the School of Data and Computer Science at Sun Yat-Sen University in December 2017, advised by Prof. Liang Lin. He obtained dual PhD awards in the Department of Computing of the Hong Kong Polytechnic University in March 2019, advised by Prof. Lei Zhang. His current research interests include computer vision and machine learning. More information can be found on his personal website https://kezewang.com.

Lingbo Liu is currently a postdoctoral fellow in the Department of Computing at the Hong Kong Polytechnic University. He received his Ph.D. degree in computer science from Sun Yat-Sen University in 2020. He was a research assistant at the University of Sydney, Australia. His current research interests include machine learning and intelligent transportation systems. He has authored and coauthored more than 15 papers in top-tier academic journals and conferences.

Haoyuan Lan received her bachelor's degree in Intelligence Science and Technology from Central South University in 2020. She is currently a master student in the School of Computer Science and Engineering, Sun Yat-Sen University. Her research interests include computer vision and deep learning.

Liang Lin (M'09, SM'15) is a Full Professor of computer science at Sun Yat-Sen University. He served as the Executive Director and Distinguished Scientist of SenseTime Group from 2016 to 2018, leading the R&D teams for cutting-edge technology transferring. He has authored or co-authored more than 200 papers in leading academic journals and conferences (e.g., 20+ papers in TPAMI/IJCV), and his papers have been cited by more than 16,000 times. He is an associate editor of IEEE Trans.Neural Networks and Learning Systems and IEEE Trans. Human-Machine Systems, and served as Area Chairs for numerous conferences such as CVPR, ICCV, SIGKDD and AAAI. He is the recipient of numerous awards and honors including Wu Wen-Jun Artificial Intelligence Award, the First Prize of China Society of Image and Graphics, ICCV Best Paper Nomination in 2019, Annual Best Paper Award by Pattern Recognition (Elsevier) in 2018, Best Paper Dimond Award in IEEE ICME 2017, Google Faculty Award in 2012. His supervised PhD students received ACM China Doctoral Dissertation Award, CCF Best Doctoral Dissertation and CAAI Best Doctoral Dissertation. He is a Fellow of IET.