Self-Supervised Video Representation Learning by Uncovering Spatio-temporal Statistics

Jiangliu Wang*, Jianbo Jiao*, *Member, IEEE*, Linchao Bao, Shengfeng He, *Senior Member, IEEE*, Wei Liu, *Senior Member, IEEE* and Yun-hui Liu, *Fellow, IEEE*

Abstract—This paper proposes a novel pretext task to address the self-supervised video representation learning problem. Specifically, given an unlabeled video clip, we compute a series of spatio-temporal statistical summaries, such as the spatial location and dominant direction of the largest motion, the spatial location and dominant color of the largest color diversity along the temporal axis, *etc.* Then a neural network is built and trained to yield the statistical summaries given the video frames as inputs. In order to alleviate the learning difficulty, we employ several spatial partitioning patterns to encode rough spatial locations instead of exact spatial Cartesian coordinates. Our approach is inspired by the observation that human visual system is sensitive to rapidly changing contents in the visual field, and only needs impressions about rough spatial locations to understand the visual contents. To validate the effectiveness of the proposed approach, we conduct extensive experiments with four 3D backbone networks, *i.e.*, C3D, 3D-ResNet, R(2+1)D and S3D-G. The results show that our approach outperforms the existing approaches across these backbone networks on four downstream video analysis tasks including action recognition, video retrieval, dynamic scene recognition, and action similarity labeling. The source code is publicly available at: https://github.com/laura-wang/video_repres_sts.

Index Terms—Self-Supervised Learning, Representation Learning, Video Understanding, 3D CNN.

1 Introduction

Powerful video representation serves as a foundation for solving many video content analysis and understanding tasks, such as action recognition [1], [2], video retrieval [3], [4], video captioning [5], [6], *etc.* Various network architectures [1], [7], [8] are designed and trained with massive human-annotated video data to learn video representation for individual tasks. While great progresses have been made, supervised video representation learning is impeded by two major obstacles: (1) Annotation of video data is labour-intensive and expensive, restricting supervised learning to relish a large quantity of free video resources on the Internet. (2) Representation learned from labeled video data lacks generality and robustness, *e.g.*, video features learned for action recognition do not well to video retrieval task [9], [10].

To tackle the aforementioned challenges, multiple approaches [11], [12], [13], [14] have emerged to learn more generic and robust video representation in a self-supervised manner. Neural networks are first pre-trained with unlabeled videos using *pretext tasks*, where supervision signals are derived from input data without human annotations. Then the learned representation can be employed as weight initialization for training models or be directly used as features in succeeding *downstream tasks*.

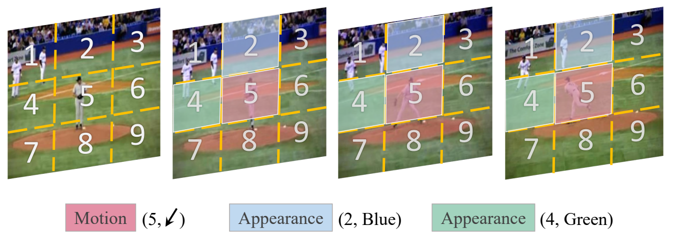


Fig. 1. Main idea of the proposed approach. Given a video sequence, we design a pretext task to uncover the summaries derived from spatio-temporal statistics for self-supervised video representation learning. Specifically, each video frame is first divided into several spatial regions using different partitioning patterns like the grid shown in the figure. Then the derived statistical labels, such as *the region with the largest motion and its direction* (the red patch), *the most diverged region in appearance and its dominant color* (the blue patch), and *the most stable region in appearance and its dominant color* (the green patch), are employed as supervision signals to guide the representation learning.

Among the existing self-supervised video representation learning methods, video order verification/prediction [11], [12], [13], [14], [15] is one of the most popular pretext tasks. It randomly shuffles video frames and asks a neural network to predict whether the video is perturbed or to rearrange the frames in a correct chronological order. By utilizing the intrinsic temporal characteristics of videos, these pretext tasks have been shown useful for learning high-level semantic features. Other approaches include flow fields prediction [16], future frame prediction [17], [18], [19], dense predictive coding [20], *etc.* Although promising results have been achieved, the dense prediction pretext tasks

- *J. Wang and Y. Liu are with the Chinese University of Hong Kong (CUHK), the CUHK T Stone Robotics Institute, and the Hong Kong Centre for Logistics Robotics. Email: {jiangliuwang@link.cuhk.edu.hk, yhliu@cuhk.edu.hk}.*
- *J. Jiao is with the Department of Engineering Science, University of Oxford. Email: jianbo@robots.ox.ac.uk.*
- *L. Bao and W. Liu are with Tencent AI Lab. Email: {linchaobao@gmail.com, wl2223@columbia.edu}*
- *S. He is with the School of Computer Science and Engineering, South China University of Technology. Email: shengfenghe7@gmail.com.*
- *∗ J. Wang and J. Jiao contributed equally to this work. L. Bao and Y. Liu are the corresponding authors.*

may lead to redundant feature learning towards solving the pretext task itself, instead of learning generic representative features for downstream video analysis tasks. For example, predicting the future frame requires the network to precisely estimate each pixel in each frame in a video clip. This increases the learning difficulties and causes the network to waste a large portion of capacity on learning features that may be not transferable to high-level video analysis tasks.

In this paper, enlightened by the human visual system [21], we propose a novel pretext task to learn video representation by uncovering spatio-temporal statistical summaries from unlabeled videos. For instance, given a video clip, the network is encouraged to identify the largest moving area with its corresponding motion direction, as well as the most rapidly changing region in appearance with its dominant color. This idea is inspired by the cognitive study on human visual system [21], in which the representation of motion is found to be based on a set of learned patterns. These patterns are encoded as sequences of "snapshots" of body shapes by neurons in the *form pathway*, and by sequences of complex optic flow patterns in the *motion pathway*. In our work, these two pathways are defined as the appearance branch and motion branch, respectively. In addition, we define and extract several abstract statistical summaries accordingly, which is also inspired by the biological hierarchical perception mechanism [21].

We design several spatial partitioning patterns to encode each spatial location and its spatio-temporal statistics over multiple frames, and use the encoded vectors as supervision signals to train the neural network for spatio-temporal representation learning. The novel objectives are informative for the motion and appearance distributions in videos, *e.g.*, the spatial locations of the most dominant motions and their directions, the most consistent and the most diverse colors over a certain temporal cube, *etc*. An illustration of the main idea is shown in Fig. 1, where a 3×3 grid pattern with motion and appearance statistics is shown for example. We conduct extensive experiments with 3D convolutional neural networks (CNNs) to validate the effectiveness of the proposed approach. The experimental results show that, compared with training from scratch, pre-training using our approach demonstrates a large performance gain for video action recognition problem (*e.g.*, 56.0% *vs.* 77.8% on UCF101 and 22.0% *vs.* 40.7% on HMDB51). By transferring the learned representation to other video tasks, such as video retrieval, dynamic scene recognition, and action similarity labeling, we further demonstrate the generality and robustness of the video representation learned by the proposed approach.

A preliminary version of this work was presented in [22], where the basic idea of utilizing spatio-temporal statistical information for video representation learning is introduced. In this paper, we further extend the previous work in five aspects: (1) We provide a more detailed implementation of the proposed self-supervised learning approach. We extend the proposed method to four backbone networks, *i.e.*, C3D with BN, 3D-ResNet, R(2+1)D, and S3D-G, on a large-scale dataset kinetics-400 [23]. (2) We conduct extensive ablation studies on the effectiveness of the pre-training dataset, the effectiveness of different bakcbone networks, and the correlation between pretext task and downstream task performances. (3) We investigate the effectiveness of different training targets, including 1D-label regression, 2D-label regression, and classification. (4) A curriculum learning strategy is introduced to further improve the representation learning. (5) We further validate the proposed method on a new downstream task, video retrieval, to evaluate the generalizability of the learned representation.

To summarize, the main contributions of this work are four-fold: (1) We propose a novel pretext task for video representation learning by uncovering motion and appearance statistics without human annotated labels. (2) We introduce a curriculum learning strategy based on the proposed spatio-temporal statistics, which is also inspired by the human learning process: from simple samples to difficult samples. (3) Extensive ablation studies are conducted and analyzed to reveal several insightful findings for self-supervised learning, including the effectiveness of training data scale, network architectures, correlation between pretext task and downstream tasks, and feature generalization, to name a few. (4) The proposed approach significantly outperforms previous approaches across all the studied network architectures in various video analysis tasks. Code and models are made publicly available online.

2 Related Work

In this section, we review related works, including self-supervised representation learning and its applications on downstream video analysis tasks. Please refer to a recent survey [24] for more details.

2.1 Self-supervised Representation Learning

Self-supervised Image Representation Learning. Self-supervised visual representation learning is first proposed and investigated in the image domain [25]. Various novel pretext tasks have been proposed to learn image representation from unlabeled image data, including predicting image context [25], re-ordering perturbed image patches [26], colorizing grayscale images [27], inpainting missing regions [28], counting virtual primitives [29], classifying image rotations [30], predicting image labels obtained using a clustering algorithm [31], *etc*. There are also studies that try to learn image representation from unlabeled video data. Wang and Gupta [32] proposed to derive supervision signals from unlabeled videos using traditional tracking algorithms. Instead Pathak *et al*. [33] obtained supervision signals using conventional motion segmentation algorithms.

Very recently, contrastive learning has yielded remarkable performance [34], [35] and attracted wide attention in the self-supervised representation learning field. The intuition behind this approach is to conduct instance discrimination. It simultaneously minimizes the distances of positive pairs, and maximizes the distances of negative pairs in the latent space. Compared with the hand-crafted pretext tasks described above, this framework allows more flexibility of the design of self-supervised learning approaches. A line of works [36], [37], [38], [39], [40] are thereby encouraged and introduced to resolve the most fundamental problem of contrastive learning: how to define the positive/negative pairs. Inspired by the empirical success, some works [41],

[42] have also made efforts to reveal the essential nature of contrastive learning in a more theoretical way. Readers are encouraged to view the self-supervised representation learning from both hand-crafted pretext tasks perspective and contrastive learning perspective, which provides a bigger picture.

Self-supervised Video Representation Learning. Inspired by the success of self-supervised image representation learning, many works have emerged to learn transferable representation for video-related downstream tasks, such as action recognition, video retrieval, *etc*. Intuitively, a large number of studies [11], [12], [13], [14], [15] leveraged the distinct temporal information of videos and proposed to use frame sequence ordering as their pretext tasks. Wei *et al*. [43] also proposed to predict whether the video clips are playing forwards or backwards. Büchler *et al*. [9] further used deep reinforcement learning to design a sampling permutations policy. Gan *et al*. [16] proposed a geometry-guided network that forces the CNN to predict flow fields or disparity maps between two consecutive frames.

Although these work demonstrated the effectiveness of self-supervised representation learning with unlabeled videos and showed impressive performances when transferring the learned features to video recognition tasks, their approaches are only applicable to a CNN that accepts one or two frames as inputs and cannot be applied to network architectures that are suitable for spatio-temporal representation. To address this problem, works [10], [13], [14] have been introduced to use 3D CNNs as backbone networks for spatio-temporal representation learning. Naturally, the 2D frame ordering pretext tasks are extended to 3D video clip ordering. Some recent works [44], [45], [46], [47] also demonstrated that predicting the video playback pace/speed is a simple-yet-effective pretext task. Inspired by the success of contrastive learning in the image domain, some works [20], [48], also attempted to extend the concept of contrastive learning in the video domain. Another line of research should be mentioned is to leverage multi-modality sources, *e.g.*, video-audio [49], [50] and video-text [51], for self-supervised representation learning. Note that in this paper, we focus on single modality, *i.e.*, only consider learning representation in the video domain.

2.2 Representation Learning for Video Analysis Tasks

Representation learning serves as a fundamental building block in tackling most video analysis tasks, such as complex action recognition [52], action detection and localization [53], [54], [55], video captioning [5], [6], *etc*. Two types of application modes are commonly adopted to evaluate the self-supervised video representation learning, either through transfer learning (as an initialization model) or feature learning (as a feature extractor).

Action recognition is one of the most widely used downstream video analysis tasks. At the beginning, researchers have developed various spatio-temporal descriptors for video representation to tackle this problem [56], [57], [58]. Promising results were achieved by improved dense trajectories (iDT) descriptors [58], the best-performing hand-crated feature. Recently, extensive efforts have been focusing on the deep neural networks development due to the impressive success achieved by CNN. Tran *et al*. [59] proposed C3D that extends the 2D kernels to 3D kernels to capture spatio-temporal video representation. Simonyan and Zisserman [7] proposed a two-stream network that extracts spatio and temporal features on RGB and optical flow inputs, respectively. Stemmed from these two works, various network architectures are designed to learn video representation, including P3D [60], I3D [1], R(2+1)D [8], *etc*. In this work, we consider to use three backbone networks, C3D [59], 3D-ResNet [8] and R(2+1)D [8] to validate the proposed approach, following previous works [10], [14]. Backbone networks pre-trained with the proposed spatio-temporal statistics will be used as weight initialization and fine-tuned on UCF101 [61] and HMDB51 [62] datasets for the action recognition downstream task.

The other kind of evaluation mode is to use the pre-trained networks as feature extractors for the downstream video analysis tasks, such as video retrieval [10], [11], [14], dynamic scene recognition [16], [63], *etc*. Without fine-tuning, such a mode can directly evaluate the generality and robustness of the learned features. Performances of the self-supervised methods are compared with both competitive hand-crated video features, such as spatio-temporal interest points [56], HOG3D [57], slow feature analysis [64], bags of spacetime energies [65], *etc*., and other self-supervised learning methods.

3 Our Proposed Approach

In this section, we first explain the high-level ideas and motivations for designing our novel pretext task with a simple illustration in Section 3.1. Next, we formally define the computation of the spatio-temporal statistical labels from the motion aspect in Section 3.2 and appearance aspect in Section 3.3. A curriculum learning strategy is presented in Section 3.4. Finally, we summarize the whole learning framework with 3D CNNs in Section 3.5.

3.1 Motivation

Inspired by human visual system, we break the process of video contents understanding into several questions and encourage a CNN to answer them accordingly: (1) Where is the largest motion in a video? (2) What is the dominant direction of the largest motion? (3) Where is the largest color diversity and what is its dominant color? (4) Where is the smallest color diversity, *e.g.*, the potential background of a scene, and what is its dominant color? The motivation behind these questions is that the human visual system [21] is sensitive to large motions and rapidly changing contents in the visual field, and only needs impressions about rough spatial locations to understand the visual contents. We argue that a good pretext task should be able to capture necessary representation of video contents for downstream tasks, while does not waste model capacity on learning too detailed information that is not transferable to other downstream tasks. To this end, we design our pretext task as learning to answer the above questions with only rough spatio-temporal statistical summaries, *e.g.*, for spatial coordinates we employ several spatial partitioning patterns to encode rough spatial locations instead of exact spatial Cartesian coordinates. In the following, we use a simple illustration to explain the basic idea.

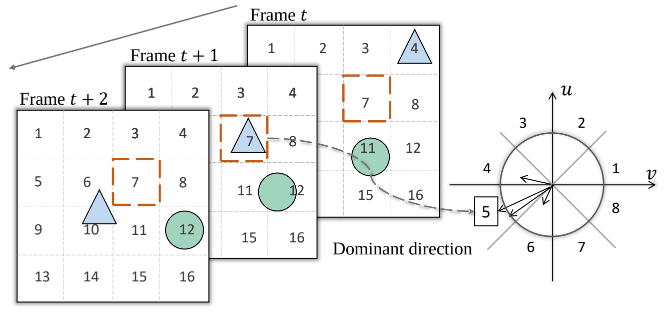


Fig. 2. Illustration of extracting statistical labels in a three-frame video clip. Detailed explanation is in Section 3.1.

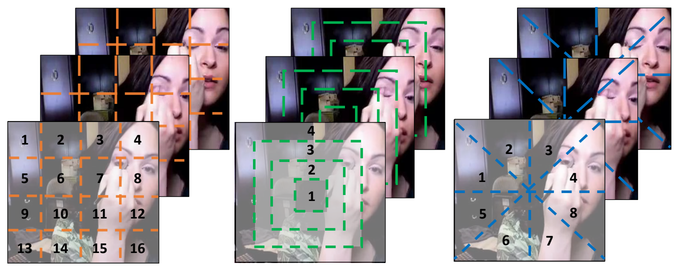


Fig. 3. Three different partitioning patterns used to divide video frames into different spatial regions. Each spatial block is assigned with a number to represent its location.

Fig. 2 shows an example of a three-frame video clip with two moving objects (blue triangle and green circle). A typical video clip usually contains much more frames while here we use the three-frame clip example for better understanding. To roughly represent the location and quantify "where", each frame is divided into 4×4 blocks and each block is assigned to a number in an ascending order starting from 1 to 16. The blue triangle moves from block 4 to block 7, and the green circle moves from block 11 to block 12. By comparing the moving distances, we can easily find that the motion of the blue triangle is larger than the motion of the green circle. The largest motion lies in block 7 since it contains moving-in motion between frames t and $t+1$, and moving-out motion between frames $t+1$ and $t+2$. Regarding the question *"what is the dominant direction of the largest motion?"*, it can be easily observed that in block 7, the blue triangle moves towards lower-left. To quantify the directions, the full angle of 360° is divided into eight pieces, with each piece covering a 45° motion direction range, as shown on the right side in Fig. 2. Similar to location quantification, each angle piece is assigned to a number in an ascending order counterclockwise. The corresponding angle piece number of "lower-left" is 5.

The above illustration explains the basic idea of extracting statistical labels for motion characteristics. To further consider appearance characteristics *"where is the largest color diversity and its dominant color?"*, both block 7 and block 12 change from the background color to the moving object color. When considering that the area of the green circle is larger than the area of the blue triangle, we can tell that the largest color diversity location lies in block 12 and the dominant color is green.

Keeping the above ideas in mind, we next formally describe the approach to extract spatio-temporal statistical labels for the proposed pretext task. We assume that by training a spatio-temporal CNN to predict the motion and appearance statistics mentioned above, better spatio-temporal representation can be learned, which will benefit the downstream video analysis tasks consequently.

3.2 Motion Statistics

Optical flow is a commonly used feature to represent motion information in many action recognition methods [1], [7]. In the self-supervised learning paradigm, predicting optical flow between every two consecutive frames is leveraged as a pretext task to pre-train the deep model [16]. Here we also leverage optical flow estimated from a conventional non-parametric coarse-to-fine algorithm [66] to derive the motion statistical labels that are predicted in our approach.

However, we argue that there are two main drawbacks when directly using dense optical flow to compute the largest motion in our pretext task: (1) optical flow based methods are prone to being affected by camera motion, since they represent the absolute motion [67], [68]. (2) Dense optical flow contains sophisticated and redundant information for statistical labels computation, increasing the learning difficulty and leading to network capacity waste. To mitigate these influences, we seek to use a more robust and sparse feature, motion boundary [67].

Motion Boundary. Denote u and v as the horizontal and vertical components of optical flow, respectively. Motion boundaries are derived by computing the x- and y-derivatives of u and v, respectively:

$$m_u = (u_x, u_y) = (\frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}),\ m_v = (v_x, v_y) = (\frac{\partial v}{\partial x}, \frac{\partial v}{\partial y}), \quad (1)$$

where m_u and m_v is the motion boundary of u and v, respectively. As motion boundaries capture changes in the flow field, constant or smoothly varying motion, such as motion caused by camera view change, will be cancelled out. Specifically, given an N-frame video clip, $(N-1) \times 2$ motion boundaries are computed based on $(N-1)$ optical flows. Diverse video motion information can be encoded into two summarized motion boundaries by summing up all these $(N-1)$ sparse motion boundaries m_u and m_v:

$$M_u = (\sum_{i=1}^{N-1} u_x^i, \sum_{i=1}^{N-1} u_y^i),\ M_v = (\sum_{i=1}^{N-1} v_x^i, \sum_{i=1}^{N-1} v_y^i), \quad (2)$$

where M_u and M_v denotes the summarized motion boundaries on u and v, respectively.

Spatial-aware Motion Statistical Labels. Based on motion boundaries, we describe how to compute the spatial-aware motion statistical labels that describe the largest motion location and the dominant direction of the largest motion. Given a video clip, we first divide it into spatial blocks using partitioning patterns as shown in Fig. 3. Here, we introduce three simple yet effective patterns: pattern 1 divides each frame into 4×4 grids; pattern 2 divides each frame into 4 different non-overlapped areas with the same gap between each block; pattern 3 divides each frame by two center lines and two diagonal lines. Then we compute summarized motion boundaries M_u and M_v as described in Eq. (2). Motion magnitude and orientation of each pixel can be obtained by transforming M_u and M_v from the Cartesian coordinates to the polar coordinates.

We take pattern 1 for an example to illustrate how to generate the motion statistical labels, while other patterns follow the same procedure. For the *largest motion location labels*, we first compute the average magnitude of blocks 1 to 16 in pattern 1. Then we compare and find out block B with the largest average magnitude from the above 16 blocks. The index number of B is treated as the largest motion location label. Note that the largest motion locations computed from M_u and M_v can be different. Therefore, two corresponding labels are extracted from M_u and M_v, respectively.

Based on the largest motion block, we compute the *dominant orientation label*, which is similar to the computation of motion boundary histogram [67]. We divide 360° into 8 bins evenly, and assign each bin with a number to represent its orientation. For each pixel in the largest motion block, we use its orientation angle to determine which angle bin it belongs to and add the corresponding magnitude value into the angle bin. The dominant orientation label is the index number of the angle bin with the largest magnitude sum. Similarly, two orientation labels are extracted from M_u and M_v, respectively.

Global Motion Statistical Labels. We further propose global motion statistical labels that provide complementary information to the local motion statistics described above. Specifically, given a video clip, the model is asked to predict the frame index (instead of the block index) with the largest motion. To succeed in such a pretext task, the model is encouraged to understand the video contents from a global perspective.

Formally, given an N-frame video clip, motion boundaries of the i^{th} frame can be computed from Eq. (1), resulting in m_u^i and m_v^i, respectively. By casting m_u^i and m_v^i from the Cartesian coordinates to the Polar coordinates, motion magnitude and orientation of each pixel can be obtained. Denote the magnitude maps as mag_u^i and mag_v^i. Then the global motion statistical labels can be computed as follows:

$$I_u = \underset{i\in\{1,\cdots,N-1\}}{\arg\max} \sum \mathsf{mag}_u^i,\ I_v = \underset{i\in\{1,\cdots,N-1\}}{\arg\max} \sum \mathsf{mag}_v^i, \quad (3)$$

where I_u, I_v are the frame indices of the largest magnitude sum (largest motion) w.r.t. u and v.

3.3 Appearance Statistics

Spatio-temporal Color Diversity Labels. Given an N-frame video clip, we divide it into spatial video blocks by patterns described above, same as the motion statistics. For an N-frame video block, we compute the volumetric color distribution V_i in the 3D color space for the i^{th} frame. We then use the Intersection over Union (IoU) along the temporal axis to quantify the spatio-temporal color diversity as follows:

$$\mathrm{IoU} = \frac{V_1 \cap V_2 \cap \cdots \cap V_i \cdots \cap V_N}{V_1 \cup V_2 \cup \cdots \cup V_i \cdots \cup V_N}. \quad (4)$$

The largest color diversity location is the block with the smallest IoU, while the smallest color diversity location is the block with the largest IoU. In practice, we calculate the IoU on R, G, B channels separately and compute the final IoU by averaging them as follows:

$$\mathrm{IoU} = (\mathrm{IoU}^R + \mathrm{IoU}^G + \mathrm{IoU}^B)\ /\ 3, \quad (5)$$

where the 3D distribution V_i used to compute IoU^R, IoU^G, and IoU^B is generated by calculating the color histogram w.r.t. each color channel.

Dominant Color Labels. Based on the video blocks with the largest color diversity and smallest color diversity, we compute the corresponding dominant color labels. We divide the 3D RGB color space into 8 bins evenly and assign each bin with an index number. Then for each pixel in the video block, based on its RGB value, we assign a number of the corresponding color bin to it. Finally, color bin with the largest number of pixels is the label for the dominant color.

Global Appearance Statistical Labels. We also propose global appearance statistical labels to provide supplementary information. Specifically, we use the dominant color of the whole video (instead of a video block) as the global appearance statistical label. The computation method is the same as the one described above.

3.4 Motion-Aware Curriculum Learning

We further propose to leverage the curriculum learning strategy [69] to improve the learning performance. The key concept is to present the network with more difficult samples gradually. It is inspired by the human learning process and proven to be effective on many learning tasks [20], [50], [70]. Recently, Hacohen and Weinshall [71] further investigated the curriculum learning in training deep neural networks and proposed two fundamental problems to be solve: (1) scoring function problem, *i.e.*, how to quantify the difficulty of each training sample; (2) pacing function problem, *i.e.*, how to feed the networks with the sorted training samples. In this work, for self-supervised video representation learning, we describe our solutions to these two problems as follows.

Scoring Function. Scoring function f defines how to measure the difficulty of each training sample. In our case, each video clip is considered to be easy or hard, based on the difficulty to figure out the block with the largest motion, *i.e.*, difficulty to predict the motion statistical labels. To characterize the difficulty, we use the ratio between magnitude sum of the largest motion block and magnitude sum of the entire videos, as the scoring function f. When the ratio is large, it indicates that the largest motion block contains the dominant action in the video and thus is easy to find out the largest motion location, *e.g.*, a man skiing in the center of a video with smooth background change. On the other hand, when the ratio is small, it indicates that the action in the video is relatively diverse or the action is less noticeable, *e.g.*, two persons boxing with another judge walking around. See Section 5.5 for more visualized examples.

Formally, given an N-frame video clip, two summarized motion boundaries M_u and M_v are computed based on Eq. (2) and the corresponding magnitude maps are denoted by M_u^{mag} and M_v^{mag}. Denote B_u, B_v as the largest motion blocks, and B_u^{mag}, B_v^{mag} as the corresponding magnitude maps. The scoring function f is defined as the maximum ratio between the magnitude sum of B_u, M_u and B_v, M_v:

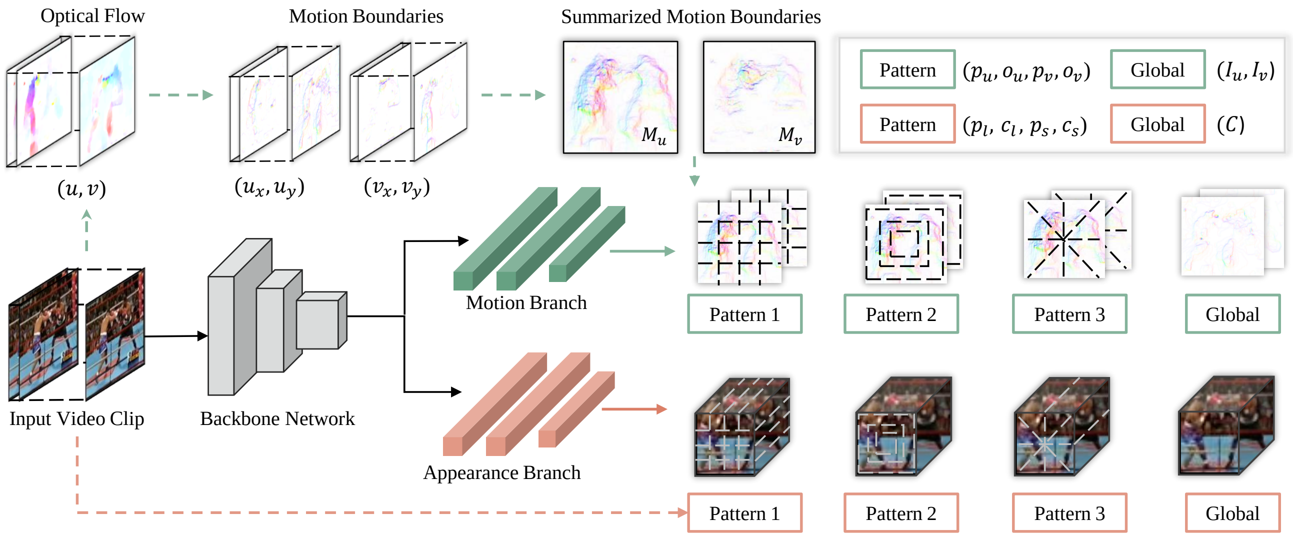


Fig. 4. Framework of the proposed approach. Given a video clip, 14 motion statistical labels and 13 appearance statistical labels are to be predicted. The motion statistical labels are computed from the summarized motion boundaries M_u and M_v. The appearance statistical labels are computed from the input video clip. For each local motion pattern, 4 ground-truth labels are generated: p_u, o_u—the spatial location of the largest magnitude based on M_u and its corresponding dominant orientation; p_v, o_v—the spatial location of the largest magnitude based on M_v and its corresponding dominant orientation. Two global motion statistical labels are I_u, I_v—the frame indices of the largest magnitude sum w.r.t. m_u and m_v. For each local appearance pattern, 4 ground-truth labels are generated: p_l, c_l—the spatial location of the largest color diversity and its corresponding dominant color; p_s, c_s—the spatial location of the smallest color diversity and its corresponding dominant color. The global appearance statistical label is C—the dominant color of the whole video.

$$f = \max\left(\frac{\sum B_u^{mag}}{\sum M_u^{mag}}, \frac{\sum B_v^{mag}}{\sum M_v^{mag}}\right). \tag{6}$$

Here we use the maximum ratio between the horizontal component u and the vertical component v as the difficulty score. The reason is that large magnitude in *one* direction can already define large motion, *e.g.*, a person running from left to right contains large motion in horizontal direction u but small motion in vertical direction v. With the scores computed from function f, training samples are sorted in a descending order accordingly, representing the difficulty from easy to hard.

Pacing Function. After sorting the samples, the remaining question is how to split these samples into different training steps. Previous works [20], [50], [70] usually adopt a two-stage training scheme, *i.e.*, training samples are divided into two categories: easy and hard. In [71], the authors formally define such a problem as a pacing function g, and introduce three stair-case functions: *single step*, *fixed exponential pacing*, and *varied exponential pacing*, where they demonstrate that these functions have comparable performances [71]. In our case, we adopt the simple single step pacing function (we also tried other functions and similarly found that they show comparable performances). Formally, we denote the first half of the training samples (descendingly sorted as aforementioned) as S_1 and the remaining half samples as S_2. Then the pacing function is defined as follows:

$$g(i) = S_1 + H(i - t) \cdot S_2, \tag{7}$$

where i is the training iteration, $H(i - t)$ is the Heaviside step function [72], and t is the number of iterations when the remaining half samples S_2 are included for pre-training. In practice, the entire S will then be used for the second-stage training when the model is converged on the first half training samples.

3.5 Learning with Spatio-temporal CNNs

The framework of our approach is illustrated in Fig. 4. Given an input video clip, a neural network is trained to uncover motion and appearance statistics defined above. Specifically, we consider C3D [59], 3D-ResNet [73], R(2+1)D [8], and S3D-G [74] as *Backbone Networks*. Regarding training targets, the proposed task could be modeled as either a regression problem or a classification problem. The effectiveness of different backbone networks and training targets is thoroughly analyzed in Sections 5.2 and 5.6, respectively.

3.5.1 Backbone Network

C3D [59] network extends 2D convolutional kernel $k \times k$ to 3D convolutional kernel $k \times k \times k$ to operate on 3D video volumes. It contains 5 convolutional blocks, 5 max-poling layers, 2 fully-connected layers, and a soft-max layer in the end to predict action class. Each convolutional block contains 2 convolutional layers except the first two blocks. Batch normalization (BN) is also added between each convolutional layer and ReLU layer.

3D-ResNet [73] is a 3D extension of the widely used 2D architecture ResNet [75], which introduces shortcut connections that perform identity mapping of each building block. A basic residual block in 3D-ResNet (R3D) contains two 3D convolutional layers with BN and ReLU followed. Shortcut connection is introduced between the top of the block and the last BN layer in the block. Following previous work [73], we use 3D-ResNet18 (R3D-18) as our backbone network, which contains four basic residual blocks and one traditional convolutional block on the top.

R(2+1)D [8] breaks the original spatio-temporal 3D convolution into a 2D spatial convolution and a 1D temporal convolution, since 3D CNNs are computationally expensive. While preserving similar network parameters to R3D, R(2+1)D outperforms R3D on supervised video action recognition task.

S3D-G [74] builds on top of I3D [1]. While achieving comparable results with I3D, S3D-G is much more efficient by introducing three vital ideas. First, it adopts a top-heavy design, *i.e.*, uses 2D convolutions in the lower layers and 3D convolutions in the higher layers. Second, it replaces 3D convolutions with separable 2D and 1D convolutions. Third, it uses spatio-temporal feature gating.

3.5.2 Training Targets

Regression. We first model the spatio-temporal statistics prediction task as a regression problem. Specifically, we consider two different designs: 1D label regression and 2D label regression.

In the 1D label regression design, we represent each statistical label by a 1D label. In this case, 27 values, including 14 motion statistical labels and 13 appearance statistical labels, are to be regressed. In the 2D label regression design, we represent the spatial location of patterns 1 and 3 by a 2D label. For example, in pattern 1, block 1 will be represented as $(1,1)$ and block 7 will be represented as $(2,3)$. In this case, 35 values, including 18 motion statistical labels and 17 appearance statistical labels, are to be regressed. L_2-norm is leveraged as the loss function to measure the difference between predicted labels and target labels. Formally, the loss function is defined as:

$$\mathcal{L}_{reg} = \lambda_m \|\hat{y}_m - y_m\|_2 + \lambda_a \|\hat{y}_a - y_a\|_2, \tag{8}$$

where $\hat{y}_m$, y_m denote the predicted and target motion statistical labels, and $\hat{y}_a$, y_a denote the predicted and target appearance statistical labels. λ_m and λ_a are the weighting parameters that are used to balance the two loss terms.

Classification. We further consider modeling our pretext task as a classification problem. Each statistical label will be predicted independently by a fully-connected layer with a cross-entropy loss. In this case, it will introduce 27 fully-connected layers. Although optimizing through such a large number of fully-connected layer seems to bias the learning towards these layers instead of the preceding layers, our experimental analysis (Section 5.6) reveals some different findings. Formally, the loss function is defined as:

$$\mathcal{L}_{cls} = \lambda_m \sum_{i=1}^{n_m} \mathcal{L}^i_{m\_cls} + \lambda_a \sum_{j=1}^{n_a} \mathcal{L}^j_{a\_cls}, \tag{9}$$

where n_m and n_a denote the numbers of motion and appearance statistical labels ($n_m = 14$ and $n_a = 13$ in our paper). Specifically,

$$\begin{aligned} \mathcal{L}^i_{m\_cls} &= -\sum_{c=1}^{M_i} y^m_{o,c} \log(p^m_{o,c}), \\ \mathcal{L}^j_{a\_cls} &= -\sum_{c=1}^{M_j} y^a_{o,c} \log(p^a_{o,c}), \end{aligned} \tag{10}$$

where $\mathcal{L}^i_{m\_cls}$, $\mathcal{L}^j_{a\_cls}$ denote the i^{th} motion and j^{th} appearance classification losses, M_i, M_j denote the number of classes for the i^{th} motion and j^{th} appearance statistical labels, $y^m_{o,c}$, $y^a_{o,c}$ indicate whether the predicted label c is the same as the target label o, and $p^m_{o,c}$, $p^a_{o,c}$ denote the predicted probabilities. λ_m and λ_a are the weighting parameters that are used to balance the two loss terms.

4 Experimental Setup

4.1 Datasets

We conduct extensive experimental evaluations on five datasets in the following sections.

Kinetics-400 (K-400) [23] contains around 306k videos of 400 action classes. It is divided into three splits: training split, validation split, and testing split. Following previous works [20], [44], [47], we use the training split as pre-training dataset, which contains around 240k video samples.

UCF101 [61] is a widely used dataset which contains 13,320 video samples of 101 action classes. It is divided into three splits. Following previous works [14], [20], [47], we use the *training split 1* as pre-training dataset and the *training/testing split 1* for downstream task evaluation.

HMDB51 [62] is a relatively small action dataset which contains around 7,000 videos of 51 action classes. Following previous works [14], [20], [47], we use the *training/testing split 1* to evaluate the proposed approach.

YUPENN [63] is a dynamic scene recognition dataset which contains 420 video samples of 14 dynamic scenes. We follow the recommended leave-one-out evaluation protocol [63] when evaluating the proposed approach.

ASLAN [76] is a video dataset focusing on the action similarity labeling problem and contains 3,631 video samples of 432 classes. During testing, following previous work [76], we use a 10-fold cross validation with leave-one-out evaluation protocol.

4.2 Implementation Details

Self-supervised Pre-training Stage. When pre-training on UCF101 dataset, video samples are first split into non-overlapped 16 frame video clips and are randomly selected during pre-training. When pre-training on K-400, following previous works [13], [20], we randomly select a consecutive 16-frame video clip and the corresponding 15-frame optical flow clip from each video sample. Each video clip is reshaped to spatial size of 128 × 171. As for data augmentation, we randomly crop the video clip to 112 × 112 and apply random horizontal flip for the entire video clip. Weights of motion statistics λ_m and appearance statistics λ_a are empirically set to be 1 and 0.1. The batch size is set to 30. We use Stochastic Gradient Descent (SGD) as the optimizer with learning rate 5×10^{-4}, which is divided by 10 every 6 epochs and the training process is stopped at 25 epochs. Model with the lowest validation loss is used for downstream stream video analysis tasks.

Supervised Fine-tuning Stage. During the supervised fine-tuning stage, weights of convolutional layers are retained from the self-supervised pre-trained models and weights of the fully-connected layers are re-initialized. The whole network is then trained again with cross-entropy loss on action recognition task with UCF101 and HMDB51 datasets. Image pre-processing procedure and training strategy are the same as the self-supervised pre-training stage, except that the initial learning rate is changed to 3×10^{-3}.

Evaluation. For action recognition task, during testing, video clips are resized to 128×171 and center-cropped to 112×112. We consider two evaluation methods: clip accuracy and video accuracy. The clip accuracy is computed by averaging the accuracy of each clip from the testing set. The video accuracy is computed by averaging the softmax probabilities of uniformly selected clips in each video [14] from the testing set. In all of the following experiments, to have a fair comparison with previous works [10], [14], [20], we use video accuracy to evaluate our approach, apart from the ablation studies on the effectiveness of each component (Section 5.1), where we use clip accuracy to keep a consistency with our previous conference paper [22].

We further use our self-supervised pre-trained models as feature extractors on three downstream video analysis tasks: dynamic scene recognition, and action similarity labeling. More evaluation details are presented in Section 6 for every downstream task.

5 Ablation Studies and Analyses

In this section, we conduct extensive ablation studies to analyze the proposed approach. We study the effectiveness of each component in Section 5.1, the effectiveness of different backbone networks in Section 5.2, the correlation between pretext and downstream task performances in Section 5.3, the effectiveness of the pre-training dataset in Section 5.4, and the effectiveness of curriculum learning strategy in Section 5.5. All these studies are conducted on top of the 1D label regression design. Then we investigate the effectiveness of different training targets in Section 5.6.

5.1 Effectiveness of Each Component

Pattern. We study the performances of three partitioning patterns. Here, we analyze and show the performances based on the motion statistics while the appearance statistics follows the same trend. As shown in Table 1(a), all three patterns achieve comparable results. Compared to random initialization, *i.e.*, training from scratch, each pattern improves by around 8%.

Local vs. Global. We study the performances of local statistics, *where is the largest motion location?*, global statistics, *which is the largest motion frame?*, and their ensemble. As can be seen in Table 1(b), when the three local patterns are combined together, we can further get around 1.5% improvement, compared to single pattern in Table 1(a). The global statistics also serves as a useful supervision signal with an improvement of 3%. All motion statistical labels, *i.e.*, local and global statistics, achieve 57.8% accuracy on the UCF101 dataset.

Motion, RGB, and Joint Statistics. We finally analyze the performances of motion statistics, appearance statistics, and their combination in Table 1(c). Both appearance and motion statistics serve as useful self-supervised signals but the motion statistics is more powerful. We hypothesize the reason is that temporal information could be more important for action recognition task. When combining motion and appearance statistics, the action recognition accuracy can be further improved.

5.2 Effectiveness of Backbone Networks

Recently, modern spatio-temporal representation learning architectures, such as R3D-18 [73] and R(2+1)D [8], have been used to validate self-supervised video representation learning methods [10], [14]. While the performances of downstream tasks are significantly improved, this practice introduces a new variable, backbone network, which could interfere with the evaluation of the pretext task itself. In the following, we first evaluate our proposed method with these modern backbone networks in Table 2. Following that, we compare our method with recent works [10], [14] on these three backbone networks in Fig. 5.

We present the performances of different backbone networks on UCF101 and HMDB51 datasets under two settings: without per-training and with pre-training in Table 2. When there is no pre-training, baseline results are obtained by training from scratch. When there is pre-training, backbone networks are first pre-trained on UCF101 dataset with the proposed method and then used as weights initialization for the following fine-tuning. We have the following observations: (1) Significant improvement is achieved on both action recognition datasets across three backbone networks. With C3D it improves UCF101 and HMDB51 by 9.6% and 13.8%; with R3D-18 it improves UCF101 and HMDB51 by 13.6% and 12.1%; with R(2+1)D it improves UCF101 and HMDB51 by 19.5% and 15.9% remarkably. (2) Compared to C3D, R3D-18 and R(2+1)D benefit more from the self-supervised pre-training. While C3D achieves the best performance in the no pre-training setting, R(2+1)D finally achieves the highest accuracy on both datasets in the self-supervised setting. (3) The proposed method using R(2+1)D achieves better performance than using R3D-18, while with similar number of network parameters. Similar observation is also demonstrated in supervised action recognition task [8], where R(2+1)D performs better than R3D-18 on K-400 dataset.

We further compare our method with two recent proposed pretext tasks VCOP [14] and VCP [10] on these three backbone networks in Fig. 5. We have three key observations: (1) The proposed self-supervised learning method achieves the best performance across all three backbone networks on both UCF101 and HMDB51 datasets. This demonstrates the superiority of our method and shows that the performance improvement is not merely due to the usage of the modern networks. The proposed spatio-temporal statistical labels indeed drive neural networks to learn powerful spatio-temporal representation for action recognition. (2) For all three pretext tasks, R(2+1)D achieves the largest improvement on both datasets, which is similar to the observation in the above experiments. (3) No best network architecture is guaranteed for different pretext tasks. R(2+1)D achieves the best performance with our method and VCOP, while C3D achieves the best performance with VCP.

5.3 Pretext Task *vs.* Downstream Task

We show the correlation between pretext and downstream task performances in Fig. 6. Specifically, we use UCF101 training split 1 as the pre-training dataset. The pretext task performance is evaluated by the mean square error between

TABLE 1
Ablation Experiments on Spatio-temporal Statistics Component.

(a) Partitioning statistical patterns

Initialization	UCF101
Random	45.4
Motion pattern 1	53.8
Motion pattern 2	53.2
Moiton pattern 3	54.2

(b) Local and global statistics

Initialization	UCF101
Random	45.4
Motion global	48.3
Motion pattern all	55.4
Motion pattern all + global	57.8

(c) Motion and appearance statistics

Initialization	UCF101	HMDB51
Random	45.4	19.7
Appearance	48.6	20.3
Motion	57.8	30.0
Joint	58.8	32.6

TABLE 2
Evaluation of different backbone networks on UCF101 and HMDB51 datasets. When pre-training, we use our self-supervised pre-training model as weight initialization.

Experimental setup			Downstream task	
Pre-training	Backbone	#Params.	UCF101	HMDB51
✗	C3D	33.4M	**61.7**	**24.0**
✓	C3D	33.4M	69.3	34.2
✗	R3D-18	14.4M	54.5	21.3
✓	R3D-18	14.4M	67.2	32.7
✗	R(2+1)D	14.4M	56.0	22.0
✓	R(2+1)D	14.4M	**73.6**	**34.1**

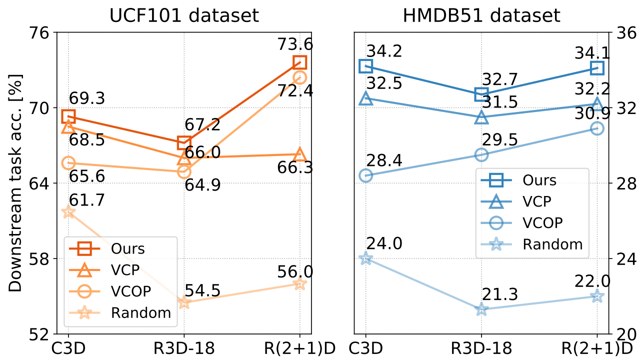


Fig. 5. Action recognition accuracy in terms of four initialization methods w.r.t three backbone networks on UCF101 and HMDB51 datasets.

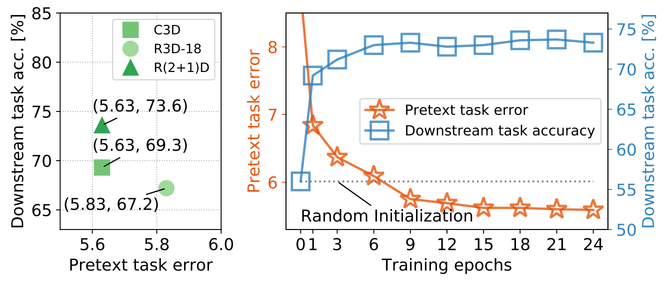


Fig. 6. Correlation between pretext task error and downstream task accuracy. Left: representative value obtained from the best performed model using different backbone networks. Right: Complete evolution using R(2+1)D as the backbone network.

the target and predicted spatio-temporal statistical labels on UCF101 testing split 1. A lower pretext task error indicates a better pretext task performance. The downstream task performance is evaluated by action recognition accuracy on the UCF101 dataset.

We have the following observations: (1) When using different backbone networks, a better pretext task performance does not guarantee a better downstream task performance. For example, in the left of Fig. 6, while C3D and R(2+1)D produce comparable pretext task error, R(2+1)D outperforms C3D by 4.3 %. (2) On the contrary, when the backbone network is fixed to R(2+1)D, as shown in the right of Fig 6, with the pretext task error decreasing, action recognition accuracy on UCF101 dataset increases. (3) The first few pre-training epochs play an important role in the downstream task performance improvement. For example, training 3 epochs can already lead to a significant improvement in the downstream task performance.

5.4 Effectiveness of Pre-training Data

In the following, we consider two scenarios to investigate the effectiveness of pre-training data. One is the comparison on different pre-training datasets with different data scales. The other is the comparison on the same pre-training dataset but with different sizes of pre-training data.

Pre-training Dataset Analysis. We analyze the performances of training on a relatively small-scale dataset UCF101 [61] and a large-scale dataset K-400 [23]. The pre-trained models are evaluated on UCF101 and HMDB51 datasets w.r.t. three different backbone networks. As shown in Fig. 7, the performance can be further improved when pre-training on a larger dataset w.r.t all the backbone networks on both downstream datasets.

Dataset Scale Analysis. We further consider pre-training networks on different proportions of *the same* K-400 dataset. In practice, $1/k$ of K-400 is used for pre-training, where $k = 16, 8, 4, 2, 4/3, 1$. To obtain the corresponding pre-training dataset, for $k = 16, 8, 4, 2$, we select one sample from every k samples of the original full K-400. As for $k = 4/3$, we first retain half of the K-400, and then select one sample from every 2 samples in the remaining half dataset. We conduct extensive experiments on three backbone networks and two downstream datasets.

As shown in Fig. 8(a), the increase of pre-training data scale does not lead to a linear increase of performance. Specifically, the increasing speed of performance is high at the beginning and then gradually decreases. Let us take R(2+1)D for example. Using $1/8$ of the K-400 can achieve half of the improvement compared to training from scratch. In addition, compared with using full K-400, using half of the K-400 only leads to *inconsequential* drop from the highest performance. When the x-axis is shown in log-scale

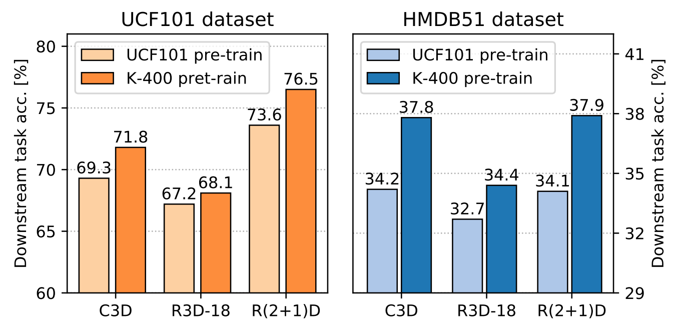
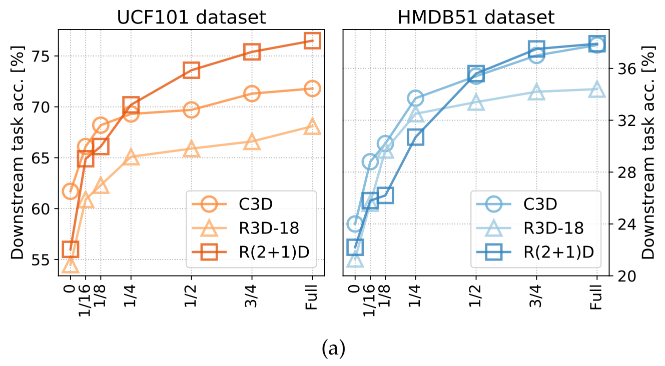


Fig. 7. Action recognition accuracy in terms of different pre-training datasets w.r.t. three backbone networks on UCF101 and HMDB51.

TABLE 3
Evaluation of curriculum learning strategy. "↑" represents the first half of the K-400 dataset while "↓" indicates the last half of the K-400 dataset.

Experimental setup		Downstream task	
Curr. Learn.	Pre-training data	UCF101	HMDB51
✗	100 % K-400	76.5	37.9
✗	50 % K-400	73.6	35.6
✗	↑, 50% K-400 (simple)	72.4	35.9
✗	↓, 50% K-400 (difficult)	72.8	32.1
✓	100% K-400	**77.8**	**40.5**



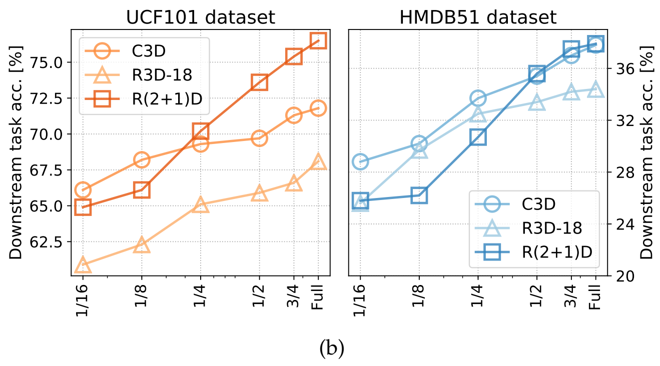


Fig. 8. Action recognition accuracy in terms of different pre-training dataset scales of K-400 shown in (a) linear-scale (position "0" indicates random initialization), and (b) log-scale w.r.t three backbone networks on UCF101 and HMDB51 datasets.

as in Fig. 8(b), with the increase of the pre-training data size, the performance of the proposed method improves *log-linearly*. Similar observations on this log-linearly improvement property are also reported in other pretext tasks of self-supervised visual representation learning [77] and in supervised learning [78], [79], [80].

5.5 Effectiveness of Curriculum Learning Strategy

We have shown that downstream task performances improve *log-linearly* with the pre-training data size in Section 5.4. Based on this observation, we suggest that it is an interesting direction to investigate the importance of different training samples in improving the downstream task performance. In this way, we can utilize the large amount of video data for self-supervised learning in a more efficient way. In the following, we show that by using the proposed curriculum learning strategy, the performance can be further improved using the same full K-400 as the pre-training dataset.

The evaluation of the proposed curriculum learning strategy is shown in Table 3. Compared with the baseline results (100% of K-400), the performances are further boosted on both UCF101 dataset and HMDB51 dataset. This validates the effectiveness of the proposed curriculum learning strategy. It is also interesting to note that when using the first or the last half of the sorted training samples, *i.e.*, simple samples or difficult samples, the performances on UCF101 dataset are both lower than the random half of K-400. Such observations further validate that the careful selection of training samples is necessary in self-supervised representation learning. Three video samples ranked from easy to hard are shown in Fig. 9.

5.6 Effectiveness of Training Targets

We evaluate the effectiveness of three different training targets: 1D label regression, 2D label regression, and classification w.r.t. three backbone networks in Table 4. Specifically, we use UCF101 training split 1 as the pre-training dataset. The pretext task performance is evaluated by spatio-temporal statistics prediction accuracy on UCF101 and HMDB51 datasets. The downstream task performance is evaluated by action recognition accuracy on UCF101 and HMDB51 datasets.

We have three key observations: (1) The proposed pretext task is quite challenging, when we formulate it as a regression problem. The mean square error loss introduces ambiguity when predicting discrete numbers. As a result, it results in a poor performance when we measure the accuracy by considering the exact number. (2) Although challenging, the proposed pretext task indeed encourages the neural networks to learn transferable representation for video understanding. The classification training target achieves the best performances on both pretext task and downstream task w.r.t. different network architectures. (3) A better pretext task performance does not always leads to a better downstream task performance, in which case too much network capacity may be allocated to the pretext task optimization. For example, the 2D label regression outperforms the 1D label regression consistently in terms of the

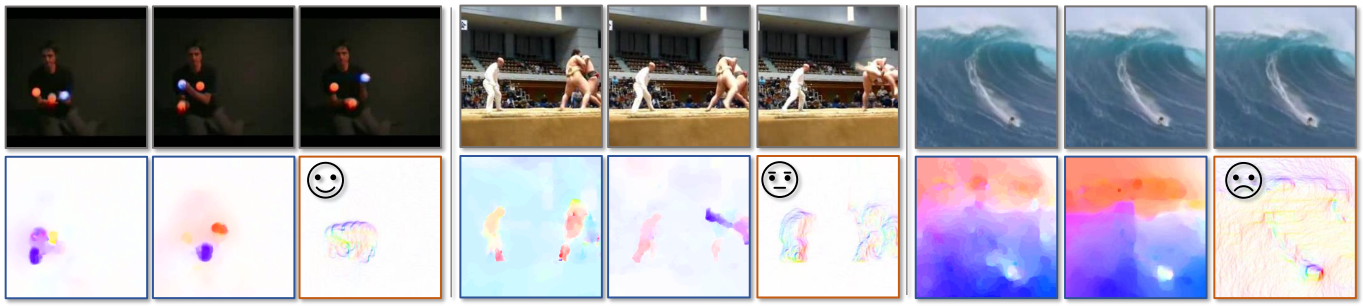


Fig. 9. Three video samples of the curriculum learning strategy. From left to right, the difficulty to predict the motion statistical labels of each video clip is increasing. For each sample, the top three images are the first, middle, and last frames of a video clip. In the bottom row, the first two images are the corresponding optical flows and the last image is the summarized motion boundaries M_u/M_v with the maximum magnitude sum.

TABLE 4
Evaluation of different training targets on UCF101 and HMDB51 datasets w.r.t three backbone networks.

	Training Target	Pretext task UCF101	Pretext task HMDB51	Downstream task UCF101	Downstream task HMDB51
C3D	1D label	20.5	19.1	69.3	34.2
	2D label	24.7	23.6	68.7	30.3
	Classification	**52.5**	**49.2**	**72.3**	**39.0**
R3D-18	1D label	18.2	17.4	67.2	32.7
	2D label	22.3	21.4	67.1	28.0
	Classification	**49.0**	**45.5**	**70.4**	**34.9**
R(2+1)D	1D label	19.6	18.2	73.6	34.1
	2D label	23.5	22.5	74.5	32.7
	Classification	**50.8**	**47.6**	**77.8**	**40.7**

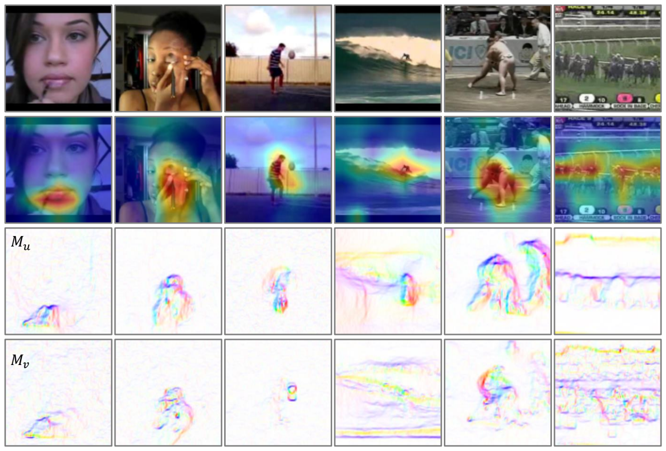


Fig. 10. Attention visualization. For each sample from top to bottom: A frame from a video clip, activation based attention map of conv5 layer on the frame by using [81], summarized motion boundaries M_u, and summarized motion boundaries M_v computed from the video clip.

pretext task performance. But it achieves worse downstream (action recognition) task performance in most cases.

6 Comparison with State-of-the-arts

In this section, we validate the proposed approach both quantitatively and qualitatively. We compare with state-of-the-arts on four downstream video understanding tasks: action recognition in Section 6.1, video retrieval in Section 6.2, dynamic scene recognition in Section 6.3, and action similarity labeling in Section 6.4.

6.1 Action Recognition

We compare our approach with state-of-the-art self-supervised learning approaches on the action recognition task in Table 5. We have the following observations: (1) Compared with random initialization, *i.e.*, training from scratch, networks fine-tuned on pre-trained models with the proposed spatio-temporal statistics (STS) achieve significant improvement on both UCF101 and HMDB51 datasets. Such results demonstrate the great potential of self-supervised video representation learning. (2) With larger input size, *i.e.*, 112 × 112 to 224 × 224, longer input length, *i.e.*, 16 frames to 64 frames, and a more powerful backbone network, *i.e.*, R(2+1)D to S3D-G, the performance of the proposed STS can be further improved drastically on both UCF101 and HMDB51 datasets. This considerably validates the scalability and potential of the proposed approach. (3) Our approach achieves state-of-the-art performance on both datasets. Given RGB videos as the network inputs, the proposed STS outperforms the state-of-the-art SpeedNet [44] by 7.9% on UCF101 and 13.2% on HMDB51. (4) The proposed approach even performs better than MemDPC* [48], which takes both RGB and optical flow as inputs and is the current best performed method; the proposed approach also achieves comparable performance with AVTS [50] and XDC [82], which use both audio and video modalities.

Attention Visualization. Fig. 10 visualizes the attention maps on several video samples using [81]. For action classes with subtle differences, *e.g.*, *Apply lipstick* and *Apply eye makeup*, the pre-trained model is sensitive to the location that is exactly the largest motion location as quantified by the summarized motion boundaries M_u and M_v. It is also interesting to note that for the *SumoWrestling* video sample (the fifth column), although three persons (two players and one judge) have large motion in direction u, only players demonstrate larger motion in direction v. As a result, the attention map is mostly activated around the players.

The performances on the action recognition downstream task strongly validate the great power of our self-supervised learning approach. The proposed pretext task is demon-

TABLE 5
Comparison with state-of-the-art self-supervised learning approaches on the action recognition task. "$*$" indicates both RGB and optical flow are taken as input, where the predictions are finally averaged.

Method	Pre-training Experimental Setup						Downstream task	
	Network	Input size	#Params	Dataset	Audio	Visual	UCF101	HMDB51
Random	R(2+1)D	224 × 224	14.4M	-		✓	56.0	22.0
Fully supervised	R(2+1)D	224 × 224	14.4M	K-400		✓	93.1	63.6
AVTS [50]	I3D	224 × 224	27.2M	K-400	✓	✓	83.7	53.0
AVTS [50]	MC3	224 × 224	11.7M	AudioSet	✓	✓	89.0	61.6
XDC [82]	R(2+1)D	224 × 224	14.4M	K-400	✓	✓	86.8	52.6
XDC [82]	R(2+1)D	224 × 224	14.4M	AudioSet	✓	✓	93.0	63.7
Object Patch [32]	AlexNet	227 × 227	62.4M	UCF101		✓	42.7	15.6
ClipOrder [11]	CaffeNet	227 × 227	58.3M	UCF101		✓	50.9	19.8
AoT [43]	AlexNet	227 × 227	62.4M	UCF101		✓	55.3	-
Deep RL [9]	CaffeNet	227 × 227	58.3M	UCF101		✓	58.6	25.0
OPN [12]	VGG	80 × 80	8.6M	UCF101		✓	59.8	23.8
VCP [10]	R(2+1)D	112 × 112	14.4M	UCF101		✓	66.3	32.2
VCOP [14]	R(2+1)D	112 × 112	14.4M	UCF101		✓	72.4	30.9
PRP [83]	R(2+1)D	112 × 112	14.4M	UCF101		✓	72.1	35.0
STS (Ours)	R(2+1)D	112 × 112	14.4M	UCF101		✓	**77.8**	**40.7**
MAS [22]	C3D	112 × 112	33.4M	K-400		✓	61.2	33.4
ST-puzzle [13]	R3D-18	224 × 224	33.6M	K-400		✓	65.8	33.7
DPC [20]	R3D-34	224 × 224	32.6M	K-400		✓	75.7	35.7
Pace [47]	R(2+1)D	112 × 112	14.4M	K-400		✓	77.1	36.6
MemDPC [48]	R-2D3D	224 × 224	32.6M	K-400		✓	78.1	41.2
TempTrans [46]	R3D-18	112 × 112	33.6M	K-400		✓	79.3	49.8
SpeedNet [44]	S3D-G	224 × 224	9.6M	K-400		✓	81.1	48.8
MemDPC* [48]	R-2D3D	224 × 224	32.6M	K-400		✓	86.1	54.5
STS (Ours)	S3D-G	224 × 224	9.6M	K-400		✓	**89.0**	**62.0**

strated to be effective in driving backbone networks to learn spatio-temporal features for action recognition. In the following, to the goal of learning generic features, we directly evaluate the learned video representation on three different downstream tasks by using the networks as feature extractors without fine-tuning on the downstream task.

6.2 Video Retrieval

We evaluate spatio-temporal representation learned from our self-supervised approach on video retrieval task. Given a video, we follow [10], [14] to uniformly sample ten 16-frame clips. Then the video clips are fed into the self-supervised pre-trained models to extract features from the last pooling layer (pool5). Based on the extracted video features, cosine distances between videos of testing split and training split are computed. Finally, the video retrieval performance is evaluated on the testing split by querying top-k nearest neighbours from the training split based on cosine distances. Here, we consider k to be 1, 5, 10, 20, 50. If the test clip class label is within the top-k retrieval results, it is considered to be successfully retrieved.

In Tables 6 and 7, we compare our approach with the other self-supervised learning methods on UCF101 dataset and HMDB51 dataset, respectively. The proposed approach outperforms VCOP [14] and VCP [10] on both datasets significantly. We further investigate whether the performances could be improved, since video features extracted from the pool5 layer tend to be more task-specific and could lack generalizability for the video retrieval task. To validate this hypothesis, we extract video features from each pooling

TABLE 6
Comparison with state-of-the-art self-supervised learning methods on the video retrieval task with the UCF101 dataset. The best results from pool5 w.r.t. each 3D backbone network are shown in **bold**. The results from pool4 on our method are in *italic* and highlighted.

	Method	Top-1	Top-5	Top-10	Top-20	Top-50
AlexNet	Jigsaw [26]	19.7	28.5	33.5	40.0	49.4
	OPN [12]	19.9	28.7	34.0	40.6	51.6
	Deep RL [9]	25.7	36.2	42.2	49.2	59.5
C3D	Random	16.7	27.5	33.7	41.4	53.0
	VCOP [14]	12.5	29.0	39.0	50.6	66.9
	VCP [10]	17.3	31.5	42.0	52.6	67.7
	Ours	**39.1**	**59.2**	**68.8**	**77.6**	**86.4**
	Ours (p4)	*43.9*	*63.4*	*71.3*	*79.0*	*87.5*
R3D-18	Random	9.9	18.9	26.0	35.5	51.9
	VCOP [14]	14.1	30.3	40.4	51.1	66.5
	VCP [10]	18.6	33.6	42.5	53.5	68.1
	Ours	**38.3**	**59.9**	**68.9**	**77.2**	**87.3**
	Ours (p4)	*42.7*	*62.3*	*71.0*	*78.3*	*87.3*
R(2+1)D	Random	10.6	20.7	27.4	37.4	53.1
	VCOP [14]	10.7	25.9	35.4	47.3	63.9
	VCP [10]	19.9	33.7	42.0	50.5	64.4
	Ours	**38.1**	**58.9**	**68.1**	**77.0**	**85.9**
	Ours (p4)	*42.2*	*63.4*	*71.3*	*78.7*	*86.9*

layer. In Fig. 11, we show the comparison between the self-supervised method (pre-trained on the proposed pretext task) and supervised method (pre-trained on the action labels) on HMDB51 dataset, and UCF101 dataset follows the similar trends.

We have the following key observations: (1) Regarding

TABLE 7
Comparison with state-of-the-art self-supervised learning methods on the video retrieval task with the HMDB51 dataset. The best results from pool5 w.r.t. each 3D backbone network are shown in **bold**. The results from pool4 on our method are in *italic* and highlighted.

	Method	Top-1	Top-5	Top-10	Top-20	Top-50
C3D	Random	7.4	20.5	31.9	44.5	66.3
	VCOP [14]	7.4	22.6	34.4	48.5	70.1
	VCP [10]	7.8	23.8	35.5	49.3	71.6
	Ours	**16.4**	**36.9**	**49.9**	**64.9**	**82.0**
	Ours (p4)	*20.7*	*36.9*	*49.9*	*64.9*	*82.0*
R3D-18	Random	6.7	18.3	28.3	43.1	67.9
	VCOP [14]	7.6	22.9	34.4	48.8	68.9
	VCP [10]	7.6	24.4	36.6	53.6	76.4
	Ours	**18.0**	**37.2**	**50.7**	**64.8**	**82.3**
	Ours (p4)	*20.1*	*42.4*	*55.6*	*68.1*	*82.3*
R(2+1)D	Random	4.5	14.8	23.4	38.9	63.0
	VCOP [14]	5.7	19.5	30.7	45.8	67.0
	VCP [10]	6.7	21.3	32.7	49.2	73.3
	Ours	**16.4**	**36.9**	**50.5**	**65.4**	**81.4**
	Ours (p4)	*19.7*	*41.8*	*55.5*	*68.4*	*83.6*

our self-supervised method, with the evaluation layer going deeper, the retrieval performance would increase to a peak (usually at pool3 or pool4 layer) and then decrease. Similar observation is also reported in self-supervised image representation learning [84]. The corresponding performance of pool4 layer is reported in Tables 6 and 7 (highlighted in blue). (2) Our self-supervised method significantly outperforms the supervised method, especially at deeper layers. This suggests that features learned from our self-supervised method are more robust and generic when transferring to the video retrieval task. Some qualitative video retrieval results are shown in Fig. 12.

6.3 Dynamic Scene Recognition

We further study the transferability of the learned features on dynamic scene recognition problem with the YUPENN dataset [63], which contains 420 video samples of 14 dynamic scenes. Following previous work [59], each video sample is first split into 16-frame clips with 8 frames overlapped. Then the spatio-temporal feature of each clip is extracted based on the self-supervised pre-trained models from pooling layers. In practice, similar to Section 6.2, we investigate the best-performing pooing layer w.r.t. each backbone network in such a problem. The best-performing layer for these three networks is *pool4*. Next, video-level representation is obtained by averaging the corresponding video-clip features, followed by L_2 normalization. Finally, a linear SVM is used for classification and we follow the same leave-one-out evaluation protocol as described in [63]. We compare our approach with state-of-the-art hand-crafted features and the other self-supervised learning methods in Table 8. The proposed approach significantly outperforms the state-of-the-art Geometry [16] by 8.1%, 6.0%, and 7.4% w.r.t. C3D, R3D-18, and R(2+1)D backbone networks, respectively.

TABLE 8
Comparison with state-of-the-art hand-crafted methods and self-supervised representation learning methods on the dynamic scene recognition task.

Method	Hand-crafted	Self-supervised	YUPENN
SOE [63]	✓		80.7
SFA [64]	✓		85.5
Object Patch [32]		✓	70.5
ClipOrder [11]		✓	76.7
Geometry [16]		✓	86.9
Ours, C3D		✓	**95.0**
Ours, R3D-18		✓	92.9
Ours, R(2+1)D		✓	94.3

TABLE 9
Comparison with different hand-crafted features and fully-supervised models on the ASLAN dataset.

Features	Hand-crafted	Sup.	Self-sup.	Acc.
C3D [59]		✓		78.3
P3D [60]		✓		80.8
HOF [76]	✓			56.7
HNF [76]	✓			59.5
HOG [76]	✓			59.8
Ours, C3D			✓	62.0
Ours, R3D-18			✓	61.7
Ours, R(2+1)D			✓	**62.1**

6.4 Action Similarity Labeling

In this section we introduce a challenging downstream task, action similarity labeling. The learned spatio-temporal representation is evaluated on the ASLAN dataset [76], which contains 3,631 video samples of 432 classes. Unlike action recognition task or dynamic scene recognition task that aims to predict the actual class label, the action similarity labeling task focuses on the *similarity* of two actions. That is, given two video samples, the goal is to predict whether the two samples are of the same class or not. This task is quite challenging as the test set contains *never-before-seen* actions [76].

To evaluate on the action similarity labeling task, we use the self-supervised pre-trained models as feature extractors and use a *linear* SVM for the binary classification, following [59]. Specifically, given a pair of videos, each video sample is first split into 16-frame clips with 8 frames overlapped and then fed into the network to extract features from the pool3, pool4 and pool5 layers. The video-level spatio-temporal feature is obtained by averaging the clip features, followed by L_2 normalization. After extracting three types of features for each video, we compute 12 different distances for each feature as described in [76]. The three 12 (dis-)similarities are concatenated together to obtain a 36-dimensional feature. Since the scales of distances are different, we normalize the distances separately into zero-mean and unit-variance, following [59]. A linear SVM is used for classification and we use the 10-fold leave-one-out cross validation same as [59], [76].

We compare our method with full-supervised methods and hand-crafted features in Table 9. We set a new base-

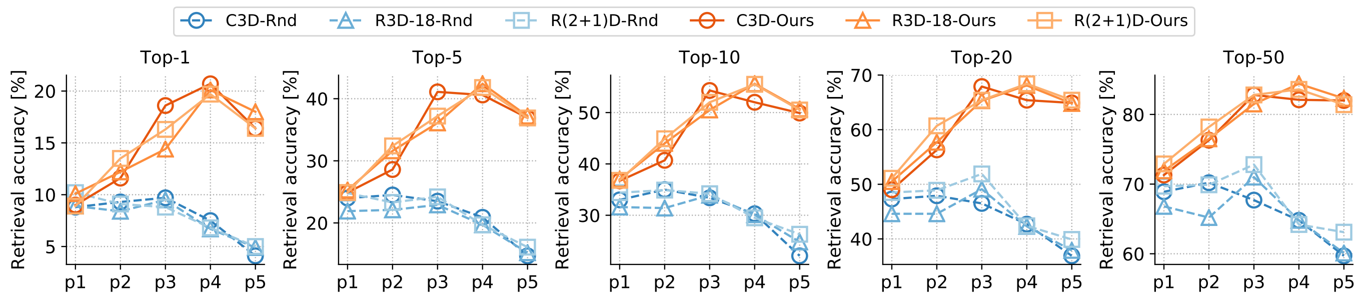


Fig. 11. Evaluation of features from different stages of the network, *i.e.*, pooling layers, on the video retrieval task with the HMDB51 dataset. The dotted blue lines show the performances of the supervised pre-trained models on the action recognition problem, *i.e.*, random initialization (Rnd). The orange lines show the performances of the self-supervised pre-trained models with our method (Ours). Better visualization with color.

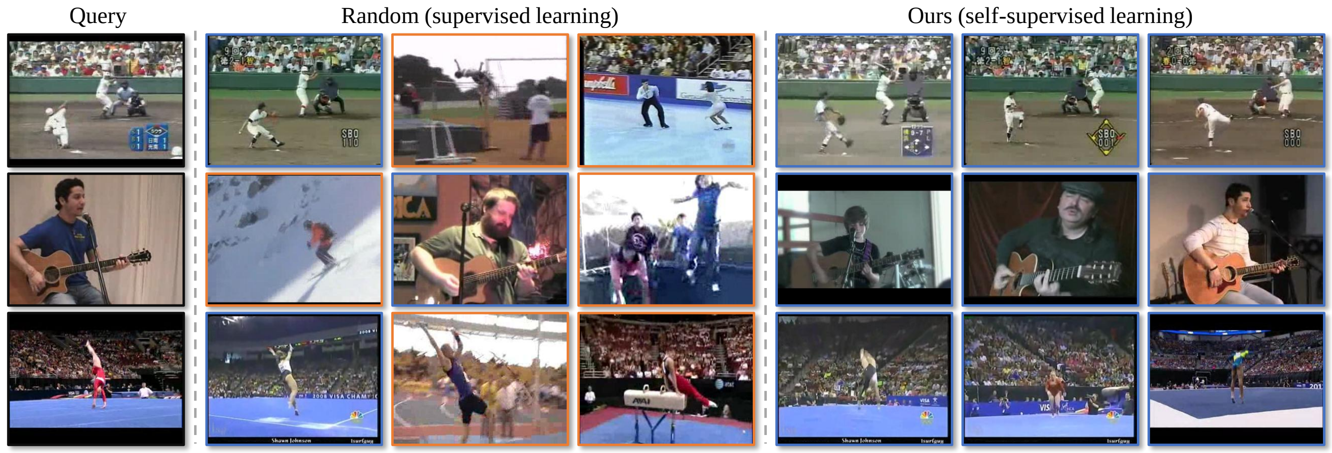


Fig. 12. Qualitative video retrieval results. From left to right: one query frame from the testing split, frames from the top-3 retrieval results based on the supervised pre-trained models, and frames from the top-3 retrieval results based on our self-supervised pre-trained models. From top to bottom: three qualitative examples of video retrieval on the UCF101 dataset. The correctly retrieved results are marked in blue while the failure cases are in orange. Better visualization with color.

line for the self-supervised method as no previous self-supervised learning methods have been validated on this task. We have the following observations: (1) Our method outperforms the hand-crafted features: HOF, HOG, and HNF (a composition of HOG and HOF). But there is still a big gap between the fully supervised method. (2) Unlike the observations in previous experiments (*e.g.*, action recognition), the performances of three backbone networks are comparable with each other. We suspect that the reason lies on the fine-tuning scheme leveraged in previous evaluation protocols, where the backbone architecture plays an important role. As a result, we suggest that the proposed evaluation on the ASLAN dataset (Table 9) could serve as a complementary evaluation task for self-supervised video representation learning to alleviate the influence of backbone networks.

7 Conclusions

In this work, we presented a novel pretext task for self-supervised video representation learning by uncovering a set of spatio-temporal labels derived from motion and appearance statistics. A curriculum learning strategy was incorporated to further improve the representation learning performance. To validate the effectiveness of our approach, we conducted extensive experiments on four downstream tasks of action recognition, video retrieval, dynamic scene recognition, and action similarity labeling, over four different backbone networks, including C3D, R3D-18, R(2+1)D, and S3D-G. Our method achieves state-of-the-art performances on various configurations. When directly evaluating the learned features by using the pre-trained model as a feature extractor, our approach demonstrates great robustness and transferability to downstream tasks and significantly outperforms the other competing self-supervised methods.

Acknowledgments

This work is partially supported by the Hong Kong RGC TRS under T42-409/18-R, the Hong Kong ITC under Grant ITS/448/16FP, the VC Fund 4930745 of the CUHK T Stone Robotics Institute, the Hong Kong Centre for Logistics Robotics, the Hong Kong-Shenzhen Innovation and Technology Research Institute (Futian), the National Natural Science Foundation of China (No. 61972162), the EPSRC Programme Grant Seebibyte EP/M013774/1 and Visual AI EP/T028572/1.

References

[1] J. Carreira and A. Zisserman, “Quo vadis, action recognition? a new model and the kinetics dataset,” in *CVPR*, 2017.

[2] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, "A closer look at spatiotemporal convolutions for action recognition," in *CVPR*, 2018.

[3] Y. Liu, S. Albanie, A. Nagrani, and A. Zisserman, "Use what you have: Video retrieval using representations from collaborative experts," *arXiv preprint arXiv:1907.13487*, 2019.

[4] A. Miech, D. Zhukov, J.-B. Alayrac, M. Tapaswi, I. Laptev, and J. Sivic, "Howto100m: Learning a text-video embedding by watching hundred million narrated video clips," in *ICCV*, 2019.

[5] B. Wang, L. Ma, W. Zhang, and W. Liu, "Reconstruction network for video captioning," in *CVPR*, 2018.

[6] J. Wang, W. Jiang, L. Ma, W. Liu, and Y. Xu, "Bidirectional attentive fusion with context gating for dense video captioning," in *CVPR*, 2018.

[7] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *NeurIPS*, 2014.

[8] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, "A closer look at spatiotemporal convolutions for action recognition," in *CVPR*, 2018.

[9] U. Buchler, B. Brattoli, and B. Ommer, "Improving spatiotemporal self-supervision by deep reinforcement learning," in *ECCV*, 2018.

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

[11] I. Misra, C. L. Zitnick, and M. Hebert, "Shuffle and learn: unsupervised learning using temporal order verification," in *ECCV*, 2016.

[12] H.-Y. Lee, J.-B. Huang, M. Singh, and M.-H. Yang, "Unsupervised representation learning by sorting sequences," in *ICCV*, 2017.

[13] D. Kim, D. Cho, and I. S. Kweon, "Self-supervised video representation learning with space-time cubic puzzles," in *AAAI*, 2019.

[14] D. Xu, J. Xiao, Z. Zhao, J. Shao, D. Xie, and Y. Zhuang, "Self-supervised spatiotemporal learning via video clip order prediction," in *CVPR*, 2019.

[15] B. Fernando, H. Bilen, E. Gavves, and S. Gould, "Self-supervised video representation learning with odd-one-out networks," in *CVPR*, 2017.

[16] C. Gan, B. Gong, K. Liu, H. Su, and L. J. Guibas, "Geometry guided convolutional neural networks for self-supervised video representation learning," in *CVPR*, 2018.

[17] C. Vondrick, H. Pirsiavash, and A. Torralba, "Generating videos with scene dynamics," in *NeurIPS*, 2016.

[18] W. Lotter, G. Kreiman, and D. Cox, "Deep predictive coding networks for video prediction and unsupervised learning," *arXiv preprint arXiv:1605.08104*, 2016.

[19] M. Mathieu, C. Couprie, and Y. LeCun, "Deep multi-scale video prediction beyond mean square error," in *ICLR*, 2016.

[20] T. Han, W. Xie, and A. Zisserman, "Video representation learning by dense predictive coding," in *ICCV Workshops*, 2019.

[21] M. A. Giese and T. Poggio, "Cognitive neuroscience: neural mechanisms for the recognition of biological movements," *Nature Reviews Neuroscience*, vol. 4, no. 3, pp. 179–192, 2003.

[22] 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 *CVPR*, 2019.

[23] 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.

[24] L. Jing and Y. Tian, "Self-supervised visual feature learning with deep neural networks: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[25] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *ICCV*, 2015.

[26] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *ECCV*, 2016.

[27] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *ECCV*, 2016.

[28] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *CVPR*, 2016.

[29] M. Noroozi, H. Pirsiavash, and P. Favaro, "Representation learning by learning to count," in *ICCV*, 2017.

[30] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *ICLR*, 2018.

[31] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *ECCV*, 2018.

[32] X. Wang and A. Gupta, "Unsupervised learning of visual representations using videos," in *ICCV*, 2015.

[33] D. Pathak, R. B. Girshick, P. Dollár, T. Darrell, and B. Hariharan, "Learning features by watching objects move." in *CVPR*, 2017.

[34] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *ICML*, 2020.

[35] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *CVPR*, 2020.

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

[37] P. Bachman, R. D. Hjelm, and W. Buchwalter, "Learning representations by maximizing mutual information across views," in *NeurIPS*, 2019.

[38] O. J. Hénaff, A. Razavi, C. Doersch, S. Eslami, and A. v. d. Oord, "Data-efficient image recognition with contrastive predictive coding," *arXiv preprint arXiv:1905.09272*, 2019.

[39] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin, "Unsupervised feature learning via non-parametric instance discrimination," in *CVPR*, 2018.

[40] Y. Tian, D. Krishnan, and P. Isola, "Contrastive multiview coding," *arXiv preprint arXiv:1906.05849*, 2019.

[41] S. Purushwalkam and A. Gupta, "Demystifying contrastive self-supervised learning: Invariances, augmentations and dataset biases," in *NeurIPS*, 2020.

[42] S. Arora, H. Khandeparkar, M. Khodak, O. Plevrakis, and N. Saunshi, "A theoretical analysis of contrastive unsupervised representation learning," in *ICML*, 2019.

[43] D. Wei, J. J. Lim, A. Zisserman, and W. T. Freeman, "Learning and using the arrow of time," in *CVPR*, 2018.

[44] 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 *CVPR*, 2020.

[45] D. Epstein, B. Chen, and C. Vondrick, "Oops! predicting unintentional action in video," in *CVPR*, 2020.

[46] S. Jenni, G. Meishvili, and P. Favaro, "Video representation learning by recognizing temporal transformations," in *ECCV*, 2020.

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

[48] T. Han, W. Xie, and A. Zisserman, "Memory-augmented dense predictive coding for video representation learning," in *ECCV*, 2020.

[49] A. Owens and A. A. Efros, "Audio-visual scene analysis with self-supervised multisensory features," in *ECCV*, 2018.

[50] B. Korbar, D. Tran, and L. Torresani, "Cooperative learning of audio and video models from self-supervised synchronization," in *NeurIPS*, 2018.

[51] A. Miech, J.-B. Alayrac, L. Smaira, I. Laptev, J. Sivic, and A. Zisserman, "End-to-end learning of visual representations from uncurated instructional videos," in *CVPR*, 2020.

[52] N. Hussein, E. Gavves, and A. W. Smeulders, "Timeception for complex action recognition," in *CVPR*, 2019.

[53] Y.-W. Chao, S. Vijayanarasimhan, B. Seybold, D. A. Ross, J. Deng, and R. Sukthankar, "Rethinking the faster r-cnn architecture for temporal action localization," in *CVPR*, 2018.

[54] Z. Shou, J. Chan, A. Zareian, K. Miyazawa, and S.-F. Chang, "Cdc: Convolutional-de-convolutional networks for precise temporal action localization in untrimmed videos," in *CVPR*, 2017.

[55] Z. Shou, D. Wang, and S.-F. Chang, "Temporal action localization in untrimmed videos via multi-stage cnns," in *CVPR*, 2016.

[56] I. Laptev, "On space-time interest points," *International Journal on Computer Vision*, vol. 64, no. 2-3, pp. 107–123, 2005.

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

[58] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *ICCV*, 2013.

[59] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *ICCV*, 2015.

[60] Z. Qiu, T. Yao, and T. Mei, "Learning spatio-temporal representation with pseudo-3D residual networks," in *ICCV*, 2017.

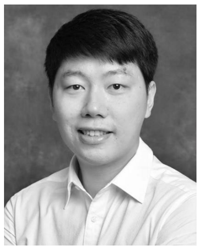
[61] K. Soomro, A. R. Zamir, and M. Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," *arXiv preprint arXiv:1212.0402*, 2012.

[62] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "Hmdb: a large video database for human motion recognition," in *ICCV*, 2011.
[63] K. G. Derpanis, M. Lecce, K. Daniilidis, and R. P. Wildes, "Dynamic scene understanding: The role of orientation features in space and time in scene classification," in *CVPR*, 2012.
[64] C. Theriault, N. Thome, and M. Cord, "Dynamic scene classification: Learning motion descriptors with slow features analysis," in *CVPR*, 2013.
[65] C. Feichtenhofer, A. Pinz, and R. P. Wildes, "Bags of spacetime energies for dynamic scene recognition," in *CVPR*, 2014.
[66] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *ECCV*, 2004.
[67] N. Dalal, B. Triggs, and C. Schmid, "Human detection using oriented histograms of flow and appearance," in *ECCV*, 2006.
[68] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, "Action recognition by dense trajectories," in *CVPR*, 2011.
[69] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *ICML*, 2009.
[70] O. Sumer, T. Dencker, and B. Ommer, "Self-supervised learning of pose embeddings from spatiotemporal relations in videos," in *ICCV*, 2017.
[71] G. Hacohen and D. Weinshall, "On the power of curriculum learning in training deep networks," in *ICML*, 2019.
[72] B. Davies, *Integral transforms and their applications*. Springer, 2002, vol. 41.
[73] K. Hara, H. Kataoka, and Y. Satoh, "Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet?" in *CVPR*, 2018.
[74] S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy, "Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification," in *ECCV*, 2018.
[75] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.
[76] O. Kliper-Gross, T. Hassner, and L. Wolf, "The action similarity labeling challenge," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 3, pp. 615–621, 2011.
[77] P. Goyal, D. Mahajan, A. Gupta, and I. Misra, "Scaling and benchmarking self-supervised visual representation learning," in *ICCV*, 2019.
[78] D. Ghadiyaram, D. Tran, and D. Mahajan, "Large-scale weakly-supervised pre-training for video action recognition," in *CVPR*, 2019.
[79] C. Sun, A. Shrivastava, S. Singh, and A. Gupta, "Revisiting unreasonable effectiveness of data in deep learning era," in *ICCV*, 2017.
[80] M. Huh, P. Agrawal, and A. A. Efros, "What makes imagenet good for transfer learning?" *arXiv preprint arXiv:1608.08614*, 2016.
[81] S. Zagoruyko and N. Komodakis, "Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer," in *ICLR*, 2017.
[82] H. Alwassel, D. Mahajan, L. Torresani, B. Ghanem, and D. Tran, "Self-supervised learning by cross-modal audio-video clustering," in *NeurIPS*, 2019.
[83] Y. Yao, C. Liu, D. Luo, Y. Zhou, and Q. Ye, "Video playback rate perception for self-supervised spatio-temporal representation learning," in *CVPR*, 2020.
[84] A. Kolesnikov, X. Zhai, and L. Beyer, "Revisiting self-supervised visual representation learning," in *CVPR*, 2019.

Jiangliu Wang received the B.E. degree from Nanjing University in 2015 and the Ph.D. degree from the Chinese University of Hong Kong (CUHK) in 2020, supported by the Hong Kong PhD Fellowship Scheme (HKPFS). She is also co-affiliated with CUHK T Stone Robotics Institute. Her research interests include video understanding, self-supervised representation learning, and related applications in robotics.

Jianbo Jiao (Member, IEEE) received the Ph.D. degree in computer science from the City University of Hong Kong in 2018, supported by the Hong Kong PhD Fellowship Scheme (HKPFS). He was a Visiting Scholar with the Beckman Institute, University of Illinois at Urbana–Champaign from 2017 to 2018. He is currently a Postdoctoral Researcher with the Department of Engineering Science, University of Oxford. His research interests include computer vision and machine learning.

Linchao Bao is currently a principal research scientist at Tencent AI Lab. He received the Ph.D. degree in Computer Science from City University of Hong Kong in 2015. Prior to that, he received M.S. degree in Pattern Recognition and Intelligent Systems from Huazhong University of Science and Technology in Wuhan, China. He was a research intern at Adobe Research from November 2013 to August 2014 and worked for DJI as an algorithm engineer from January 2015 to June 2016. His research interests include computer vision and graphics.



Shengfeng He (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees from Macau University of Science and Technology, and a Ph.D. degree from City University of Hong Kong. He is currently an Associate Professor in the School of Computer Science and Engineering, South China University of Technology. His research interests include computer vision, image processing, and computer graphics. He serves as an Associate Editor of the Neurocomputing and IEEE Signal Processing Letter.

Wei Liu (M'14-SM'19) is currently a Distinguished Scientist of Tencent, China and a director of Computer Vision Center at Tencent AI Lab. Prior to that, he has been a research staff member of IBM T. J. Watson Research Center, Yorktown Heights, NY, USA from 2012 to 2015. Dr. Liu has long been devoted to research and development in the fields of machine learning, computer vision, pattern recognition, information retrieval, big data, etc. Dr. Liu currently serves on the editorial boards of IEEE Transactions on Pattern Analysis and Machine Intelligence, IEEE Transactions on Neural Networks and Learning Systems, IEEE Transactions on Circuits and Systems for Video Technology, Pattern Recognition, etc. He is a Fellow of the International Association for Pattern Recognition (IAPR) and an Elected Member of the International Statistical Institute (ISI).

Yunhui Liu (Fellow, IEEE) received the B.E. degree in applied dynamics from the Beijing Institute of Technology, Beijing, China, the M.E. degree in mechanical engineering from Osaka University, Suita, Japan, and the Ph.D. degree in mathematical engineering and information physics from the University of Tokyo, Tokyo, Japan, in 1992. He was with the Electrotechnical Laboratory of Japan, Tsukuba, Japan, as a Research Scientist, then he joined The Chinese University of Hong Kong, Hong Kong, in 1995 and is currently a Choh-Ming Li Professor with the Department of Mechanical and Automation Engineering, the Director of the CUHK T-Stone Robotics Institute, and the Director of the Hong Kong Centre for Logistics Robotics. He has published over 200 papers in refereed journals and refereed conference proceedings. Dr. Liu was a recipient of the Highly Cited Author (Engineering) Award by Thomson Reuters in 2013 and numerous research awards from international journals and international conferences in robotics and automation and government agencies. He was the Editor-in-Chief of Robotics and Biomimetics and served as an Associate Editor of the IEEE TRANSACTIONS ON ROBOTICS AND AUTOMATION, and General Chair of IROS 2006.