博士論文

Video Representation Learning for Action Recognition and Retrieval

(行動認識・検索のための映像表現学習)

48-187421

陶 砺

指導教員　山﨑　俊彦　准教授

東京大学大学院　情報理工学系研究科　電子情報学専攻

This dissertation is submitted for the degree of
Doctor of Philosophy

December 2021

Acknowledgements

I am incredibly grateful to my supervisor, Prof. Toshihiko Yamasaki, for his continuous support, insightful advice, and patience during my doctoral study. His expertise is invaluable in research questions and methodology. And he always helps me in my academic research, as well as living in Japan. I would like to thank Aya Egawa for all her support and help with many issues at the university.

I would like to express sincere appreciation to Prof. Kiyoharu Aizawa and Prof. Yusuke Matsui for valuable advice at Aizawa-Yamasaki-Matsui Lab. Their questions based on a deep understanding of broad research areas sometimes shed light on valuable viewpoints that I would not come up with. I also thank all the current and previous lab members for fruitful discussions and for having various and funny conversations. Especially, I thank Dr. Xueting Wang, who was a post-doc in our lab, offering me thoughtful discussions and I have learned a lot from her.

I would like to gratefully acknowledge the committee members, Prof. Shin'ichi Satoh and Prof. Koiti Hashida, for their insightful comments during my preparation for the dissertation. I am very grateful to The Ministry of Education, Culture, Sports, Science and Technology - MEXT scholarship and Yamasaki Lab for their generous financial support for my Ph.D. research.

Finally, I would like to express my deepest gratitude to my parents for their everlasting love and support. Dedicated to my dear grandpa, for pleasant memories and may he rest in peace. I would give my most sincere thanks to my wife. She stands by my side, accompanies me, and encourages me to overcome difficulty all the way. Love spans around 1,777 kilometers and two time zones.

Abstract

The development of digital devices makes it much more convenient for ordinary people to create, edit, and share videos. However, many steps have to be done manually and are time-consuming to ensure high quality when processing videos because video understanding is necessary for many tasks. Many methods have been proposed to extract good videos representations and applied to video understanding tasks, such as action recognition, video retrieval, etc.

Video representation learning is the most fundamental task in video understanding. Good video representations contain sufficient information and can help with a lot of video-related downstream tasks. To obtain good representations, many recent works have required additional calculation on hand-crafted motion features, even though the computation of convolutional neural networks is already very high. And large annotated videos are necessary for specific tasks. In this thesis, we have tackled two video representation learning paradigms (i.e., supervised learning and self-supervised learning) and proposed solutions to obtain good video representations without increasing the complexity of models.

First, we address the task of supervised action recognition. We propose a new data modality with 3D convolutional neural networks, which requires stacked frames (i.e., video clips) as input data. We confirm that by simply replacing traditional RGB video clips with stacked frame differences, the network can extract better temporal information. Greater generalization ability can also be ensured when applying this kind of video representation to other video-related tasks.

Second, we propose a novel learning framework in video self-supervised learning, which can help learn good video representations without any annotations. Intra-

negative samples are generated to benefit contrastive learning. We show that by introducing negative samples by breaking the temporal relations while maintaining the spatial similarities, the network can focus more on the temporal clues, resulting in better performance when applied to the downstream video understanding tasks.

Third, we try to bridge the gap between contrastive learning and pretext tasks in video self-supervised learning. We demonstrate that a simple combination of contrastive learning and pretext tasks with proper training strategies can contribute to better video representations than that on their own. We validate the generality of this combination, explore the potential mechanism, and try to reach as closer to the performance limits of traditional video self-supervised learning methods, which are much better than corresponding baselines as reported in the original papers.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Video Representation Learning

Video understanding is the key topic across all video-related tasks, from low-level video tasks such as video enhancement, super-resolution, to high-level video tasks such as action recognition, video segmentation, video retrieval, etc. Technologies to automatically and effectively process video data boost very fast in the past decades. Learning-based methods have proved to have outstanding performance in the image research field and have been extended to videos. In recent years, deep learning methods using convolutional neural networks (CNNs) have been introduced and applied to video understanding tasks. Following the footprint in the image research field, these successful models in action recognition tasks can be used as feature extractors to encode videos and extract high-quality video representations. Some video tasks directly use these video features for further processing. To ensure that models can capture good video representations, a combination of hand-crafted features (e.g., optical flow), deeper network architectures, larger annotated datasets are also necessary, greatly increasing the complexity and difficulty throughout the workflow. We illustrate the traditional workflow for video representation learning in Fig. 1.1. The calculation of optical flow highly increases the computation complexity, and this training paradigm needs large-scale annotated video data, which requires a wealth of resources.

Video representation learning can be conducted in different ways considering the supervision signals in learning-based methods. In this thesis, we focus on both **supervised learning** and **self-supervised learning** in video representation learning, addressing the high cost for both computation and label annotation in Fig. 1.1. Tran *et al*. [6] introduced 3D convolutional networks (3D CNNs) to supervised

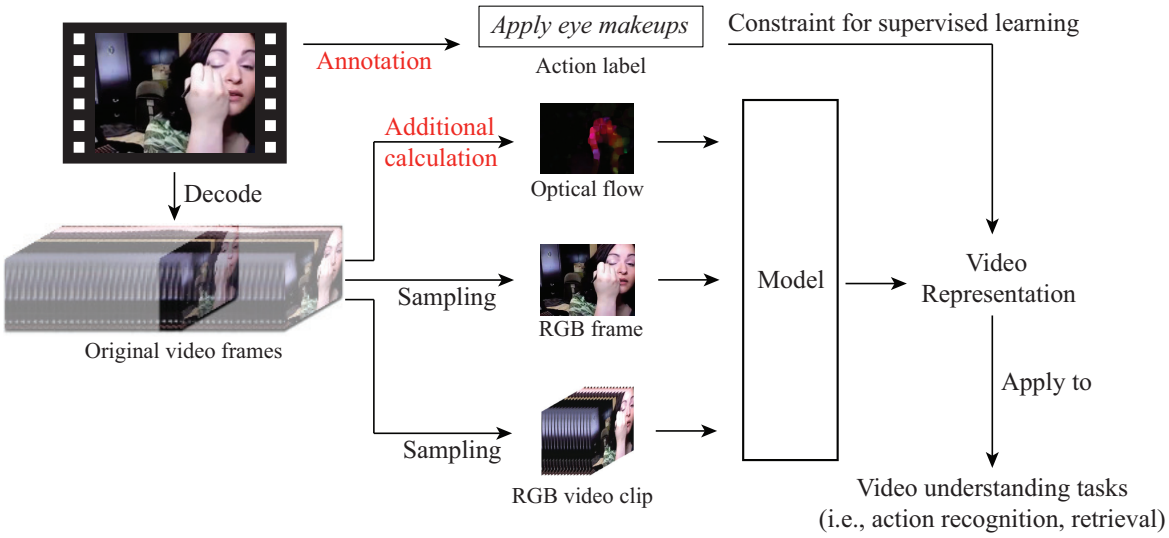


Figure 1.1: Traditional workflow for video representation learning. To ensure good performance, additional computation of optical flow and large annotated video datasets are necessary. After model optimization, the network can be used to extract video representations, which can applied to a variety number of video understanding tasks.

action recognition, which can process several frames at the same time by performing convolution along the temporal axis as well as the spatial part. This learning paradigm in action recognition tasks can constrain the model to learn good video representations, which have been validated by classification as well as clustering tasks. The success of 3D CNNs inspires a lot of following works [7–10], introducing effective network backbones from image classification to videos by replacing the 2D convolutional layers with 3D ones. With better video representation learning models, a broad range of video-related tasks enjoy the benefits and have shown great improvements such as action detection [11–14], video assessment [15, 16], and video summarization [17, 18]. All these methods still use 3D CNNs with RGB frames decoded from videos, in which condition temporal features are not well extracted. Thus, many methods [9, 19–22] pick up hand-crafted features such as optical flow information, to compensate for the loss of temporal clues in video representations. This kind of strategies further increases the complexity of the solution in both training and inference periods.

Supervised learning requires a large amount of annotated data. To make use of sufficient video data while ignoring the high cost of annotation, unsupervised learning methods have attracted more and more attention in video representation

learning. The network architectures can be directly from the corresponding supervised part, while this kind of learning paradigm can help optimize the network without any video annotations. Supervision signals are from pre-defined transformations or video sequence orders, ensuring the training is in a self-supervised learning manner. Several temporal-related pretext tasks have been proposed and by solving these temporal-related tasks [5, 23–30], the models are expected to have the ability to extract good video representations for other video tasks. Contrastive learning [2, 31–43] is another way, which try to distinguish one sample from another. Discriminative features from videos can be extracted, which are usually treated as effective feature representations.

In this thesis, the aim is to extract good video representations in an efficient and effective way. Supervised learning can help extract action features and self-supervised learning can make use of a large number of unlabeled videos, both of which are efficient solutions for video representation. To extract better temporal information without extra computation of hand-crafted features such as optical flow data, based on 3D CNNs, we build our solution with a novel data modality which is firstly introduced in a supervised manner. Taking the advantage of rich experience in supervised learning, two different approaches in self-supervised video representation learning are also proposed. The rest of the introduction chapter is organized as follows. In Sec. 1.2, we discuss challenges in video representation learning, which we should overcome. In Sec. 1.3, we present an overview of our approach for supervised video representation learning. In Sec. 1.4, we present an overview of our approach for self-supervised video representation learning.

1.2 Research Challenges

We first explain one of the most fundamental and successful approaches for video models, 3D CNNs, which are widely used and applied in both supervised learning and self-supervised learning when coping with video data. Compared to 2D convolution, which filters and aggregates spatial features and has succeeded in image tasks in computer vision, 3D convolution is naturally recognized to have excellent abilities in automatically capturing rich spatiotemporal features from videos. Videos contain T frames in resolution $H \times W$, where the channel number C is ignored. When stacking several frames together to form the input data, named as *video clips*, these video clips are in shape $T \times H \times W$. The 3D convolution kernels are also in three dimensions, with shape $K_T \times K_H \times K_W$, corresponding to the video

data. Therefore, following the success of 2D CNNs in images, 3D CNNs are widely used in different network architectures [7–10]. However, deep learning methods are usually black-box approaches, meaning that it is not sure whether the trained models have captured good spatiotemporal information, except for the performance in tasks such as action recognition tasks.

For video data, appearance information does not change too much across adjacent frames, causing an imbalance problem if treating spatial information and temporal information separately. Many researchers have realized these biases in video solutions. To address this problem, different kinds of methods have been proposed, such as increasing the depth of network architectures [7, 8, 22, 44], using larger and larger datasets for their variety in appearance across different videos [9, 45, 46], and introducing hand-crafted features [9, 19–22] which are specially designed for motion extraction. All these solutions can alleviate the drawback more or less. However, the price is high considering the computation complexity as well as the cost for data annotation. How to efficiently learn effective, general video representations remains a problem.

One of the solutions to this drawback is to obtain large and larger annotated video data. For action recognition, video segmentation, as well as other video-related tasks, the annotation formats vary from one to another. With the increasing number of videos and various application situations, it is impossible to directly apply action recognition models as feature extractors to other video research fields because of the gaps between both task domains and data domains. Self-supervised learning approaches are proposed to make use of video data without annotations. Self-supervised learning methods do not use action labels or other video-task labels. Many researchers have tried to follow the successful workflow from images to videos and achieved acceptable performance, it is still far from good representations because of the weakness in capturing temporal features. Self-supervised learning methods need to be carefully designed. Because spatial features can be easily extracted with high quality, our efforts in video representation learning focus on increasing the model abilities in extracting better temporal features.

1.3 Supervised Video Representation Learning

In the first half of the thesis, we focus on supervised action recognition tasks to explore better ways for video representation learning. Specially, we want to overcome the drawback that capturing high-quality temporal features is expensive. A lot

of works have been addressed on the network architectures. On the contrary, we are interested in the data modality part, by considering these networks do have the ability while the cause of the drawback is from the data part. We develop an approach to generate a new data modality for 3D CNNs in action recognition.

In Chapter 2, based on 3D CNNs as the network backbones, we present a new data modality in action recognition. Traditional input data (i.e., RGB video clips) are constructed by stacking RGB video frames. To force the model to focus more on temporal information, we reduce the appearance information by replacing RGB clips with stacked frame differences, named *residual clips*. This simple transformation can eliminate the side effects from the still background while maintaining compatibility with 3D CNNs, which is simple and effective. Experimental results show that it is sufficient to extract temporal features to overcome severe overfitting problems while improving the generalization ability. We prove that the success of this solution is not only in the action recognition task but also in the video retrieval task. Deep analyses based on visualization and quantitative evaluation demonstrate the effectiveness of our proposal. Because residual clips are generated from frame differences, a natural drawback is lacking some appearance information for cases where the scenes and objects play important roles. Thus, we also make use of a simple 2D CNN to compensate for the spatial information, making the solution more comprehensive without introducing complex branches.

1.4 Self-Supervised Video Representation Learning

In the latter half of the thesis, we address video representation learning in the self-supervised paradigm, while still focusing on better temporal feature extraction. The contrastive learning method is one kind of self-supervised learning approach in natural language processing and image tasks and has been applied to videos. Inspired by the design of contrastive learning, we take the advantage of temporal clues and build an inter-intra contrastive learning framework in Chapter 3. Temporal-related pretext tasks have also been proposed as another kind of solution in video self-supervised learning. We bridge the gap between these two kinds of methods to develop a joint learning framework in Chapter 4, which takes advantage of both.

In Chapter 3, we focus on the usage of contrastive learning, whose key idea is to design a task to distinguish samples from one to another. It is easy to learn discriminative features to meet the requirements of this constraint, while the quality of learned video representations is not good because directly applying contrastive

learning to videos does not take temporal information in the supervision signal. We present inter-intra contrastive (IIC) learning framework, which makes use of temporal information in an unsupervised learning way. Inter-samples mean samples come from different instances (i.e., videos) while intra-samples indicate samples are from the same instance. In addition to intra-positive and inter-negative samples which have already been used in contrastive learning, intra-negative samples are first introduced in contrastive learning. Experimental results quantitatively and qualitatively demonstrate the effectiveness of our model, indicating that our model can extract better video representations for different video tasks such as video recognition and retrieval.

In Chapter 4, we focus on the combination of pretext task-based methods and contrastive learning methods. Pretext tasks can be used to train in self-supervised learning because pre-defined tasks usually share similar feature representations, which are also useful for other related tasks. These methods are different from contrastive learning methods because pretext tasks aim at exploring effective features within samples themselves. However, good video representations should not only represent the general spatiotemporal information, but also distinctive parts among different videos to distinguish them. On the contrary, the focus of traditional contrastive learning methods is opposite, just paying attention to distinguishing videos from one to another. Inspired by the essential learning targets of these two kinds of methods, we build a joint learning framework, pretext-contrastive learning (PCL) framework, which constrains the network by both pretext task and contrastive learning. We demonstrate that with PCL framework, performance can be greatly improved over the corresponding baselines. Analyses indicate that this simple combination can take advantage of both pretext tasks and contrastive learning. Further, the generalization abilities of PCL are validated using different network backbones on different datasets in both video recognition and retrieval tasks.

Chapter 2

Residual Frames with 3D ConvNets

2.1 Introduction

For video understanding tasks such as action recognition, it is an important challenge to extract good motion representations among multiple frames. Various methods have been designed to capture the movement. 2D ConvNet based methods used interactions in the temporal axis to include temporal information [47–51]. 3D ConvNet based methods improved the recognition performance by extending 2D convolution kernel to 3D, and computations among temporal axis in each convolutional layers are believed to handle the movements [6–10, 52]. State-of-the-art methods showed further improvements by increasing the number of frames used in 3D ConvNets and the resolution of the input data as well as employing deeper backbone networks [11, 22, 44].

In a typical implementation of 3D ConvNets, these methods used stacked RGB frames as the input data. However, this kind of input is not considered enough for motion representation because the features captured from the stacked RGB frames may pay more attention to the appearance feature including background and objects rather than the movement itself, as shown in the top example in Fig. 2.1. Thus, combining with an optical flow stream is necessary to further represent the movement and improve the performance, such as the two-stream models [21, 53, 54]. However, the processing of optical flow greatly increases computation time[1]. Besides, obtaining two-stream results is possible only if the optical flow data are first extracted, which causes high latency. Frame differences have been tried in [24, 25, 48, 55]. All those methods just treated frame differences as an additional experimental modality

[1]Because there are many types of implementation of optical flow, we do not refer to any specific type of implementation. But the calculation of optical flow is generally expensive.

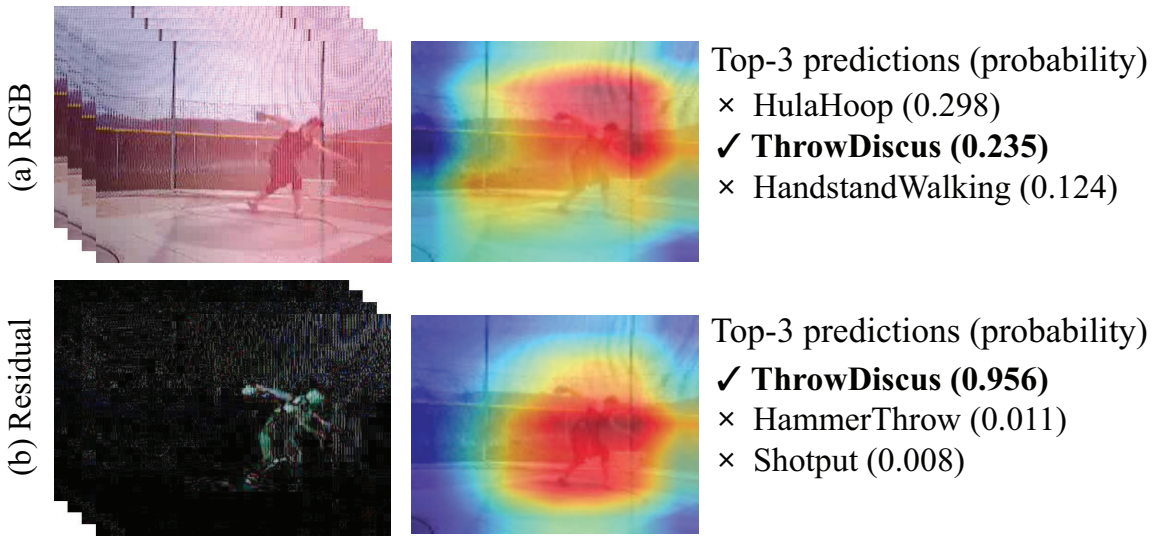


Figure 2.1: An example of our residual frames compared with normal 3D ConvNet inputs. The residual-input model focused on the movement part while RGB-input model paid more attention on background, which leads to lower accuracy for prediction.

for their networks. They did not have deep analysis except for the performance on their tasks.

Unlike most action recognition methods which focused on the network architecture, we mainly focus on the data part. In this chapter, we propose an effective strategy based on 3D convolutional networks to pre-process RGB frames to be set as the replacement of traditional input data. Our method retains what we call **residual frames**, which contain more motion-specific features by removing still objects and background information and leaving mainly the changes between frames. Through this, the movement can be extracted more clearly and recognition performance can be improved comparing to just using stacked RGB inputs as shown in the bottom sample in Fig. 2.1. One may think that our approach is naive and therefore cannot be applied to videos with global camera motion, but this will also be addressed in Section 2.5.1. Experiments reveal that our approach can yield significant improvements over top-1 accuracies when those ConvNets are trained from scratch on UCF101 [56], HMDB51 [57], and Something-something (v1 and v2) [58] datasets. Good performance of our proposal can also be easily achieved by directly fine-tuning from current existing RGB pre-trained models for only a few epochs. Models on action recognition can be directly used as a feature extractor in other video understanding tasks such as video retrieval. Our experiments also show

that residual-input model can have better performance for video retrieval on unseen datasets than RGB counterpart. We also tried to apply our residual inputs to several self-supervised learning methods [2, 3, 5], surpassing RGB baselines easily. The scratch training results on small datasets, the video retrieval performance on unseen datasets, and the successful application in self-supervised learning, showing that residual-input model have better generalization ability, will be discussed in Sec. 2.6.

For larger action recognition datasets such as Something-something (v1 and v2) [58] datasets, because they are more temporal related datasets, our residual-input model can easily achieve better performance than those methods using traditional RGB video clips as input. For other action recognition datasets such as Mini-Kinetics [10] and Kinetics [45], the definitions of the actions become more complex. For example, the category *Yoga* contains various combination of simple actions, and these datasets have a large amount of compound labels, such as *playing guitar* and *playing ukulele*, where the movement is almost the same and the difference is mainly on the objects. In this case, it is difficult to distinguish by only motion representation without enough appearance features. Therefore, when applied to these datasets, we further propose a two-path solution, which combines the residual input path with a simple 2D ConvNet to extract appearance features from a single frame in the video. Experiments show that our proposed two-path method obtains better performance over some two-stream models on UCF101 / HMDB51 / Mini-Kinetics datasets when using the same input shapes and similar or even shallower network architectures.

Our contributions are summarized as follows:

- We are the first to use and deeply analyze residual frames with 3D ConvNets for action recognition, which is simple, fast, but effective.
- Analyses including category digging, case study, network visualization, and heatmap explanations fill the gap of this kind of modality about why and when it functions well. These analyses also indicate that our proposal can extract better motion representation for actions than RGB counterparts.
- Our proposal can achieve better performance than the RGB counterparts when models are trained from scratch on four benchmark datasets with the same settings. Our results can even achieve better performance with less computation cost than methods using optical flow.
- The proposed residual-input model shows better performance in video retrieval based on several self-supervised learning methods, revealing the generalization ability and greater potential on other video understanding tasks.

We would like to clarify that we are proposing a new solution for motion representation. For this purpose, we do not always focus on the better performance than other approaches based on very deep and complex DNN architectures as well as other training / parameter settings. Instead, we discuss why and how much our approach is reasonable as compared to optical-flow-based and RGB-only approaches under the same settings.

2.2 Related Works

In this section, traditional action recognition networks are introduced first. Though temporal modeling usually exists in these networks, we use another subsection to introduce and discuss this in detail because temporal information is a key feature. Model combination is set as another subsection to clearly show the solution route maps for high accuracies.

2.2.1 Deep Action Recognition

2D Solution

2D ConvNet based methods mainly consist of frame-level feature representation and temporal modeling to fuse these features. When treating each frame of a video as a single image, 2D ConvNets which are effective for image classification task can be directly applied to video recognition. Karpathy et al. [47] tried different ways to fuse features from 2D ConvNet and then used fused features to classify videos. Temporal Segment Networks (TSN) [48] was designed to extract average features from stride sampled frames. Two-stream ConvNets [21, 53, 54] used an additional optical flow stream. And for both RGB stream and optical flow stream, 2D ConvNets were used. Recent works such as Non-local networks [51], Temporal Bilinear Networks (TBN) [49], Temporal Shift Module (TSM) [50], and Temporal Excitation and Aggregation (TEA)network [59] are variants of 2D ConvNets. Compared to 3D counterparts, 2D ConvNet based methods are more efficient because fewer parameters are used, and the performance is highly related to the temporal modeling. Our method uses a 2D network to extract appearance features considering the high efficiency of 2D models, and the proposed appearance path uses less input than existing 2D ConvNets, which is more efficient.

3D Solution

3D ConvNet based methods directly use 3D convolution kernels to process input video frames. The computation between frames is carried out when the temporal kernel size is 2 or larger, and spatial-temporal features can be automatically learned by network optimization. Tran et al. [6] proposed C3D, which consists of eight directly-connected convolutional layers and two fully-connected layers. Hara et al. [8] conducted many experiments on the 3D version of ResNet [60], including different depths and using some variants such as ResNeXt [61]. Carreira et al. [9] proposed I3D based on Inception network [62]. SlowFast [11] used two ResNet pathways to capture multi-scale information in the temporal axis. Despite different network architectures, 3D convolution kernel also has variants. One $k \times k \times k$ kernel can be separated into two parts, $k \times 1 \times 1$ and $1 \times k \times k$. Based on this, P3D [52], R(2+1)D [7], and S3D [10] were proposed. The backbones of mainstream networks are also ResNets [60] and Inception network [62]. Recently, group convolution and channel-wise convolution have been applied in 3D ConvNets [44, 63]. Neural architecture search (NAS) is used in [64] to get efficient network architectures. There are many other 3D-conv based models such as X3D [63], TPN [65], and V4D [66], which are designed to embed temporal information more effectively. However, because the parameter number is larger than 2D counterparts, 3D models are prone to over-fitting when trained from scratch on small datasets such as UCF101 [56] and HMDB51 [57]. Fine-tuning models pre-trained on very large dataset such as Kinetics [45] is one solution to acquire good performance on these small datasets. From another point of view, our proposed method focuses more on the movement itself and utilizes a 3D ConvNet with higher motion representation ability by using residual frames as input. In this way, we can reduce the tendency to over-fit on small datasets compared to standard RGB inputs when using the same network architectures.

2.2.2 Temporal Modeling

For 2D ConvNets, some models [47, 48] have been proposed which simply averaged frame features to represent videos. Donahue et al. [67] used 2D models to extract features using long short-term memory (LSTM) [68]. Zhou et al. [69] proposed Temporal Relation Network to learn temporal dependencies. Wang et al. [51] proposed non-local block to capture corresponding information among frames. Temporal Bilinear Networks [49] used temporal bilinear modeling to embed temporal

information. TSM [50] shifted 2D feature maps along temporal dimension. TEA [59] aggregated temporal features by processing information from adjacent frames in their motion excitation block.

For 3D ConvNets, temporal modeling is automatically processed by learning kernels in the temporal axis. Because 3D ConvNets use stacked RGB frames as input, the computation among frames is believed to learn motion features, while the spatial computation is for spatial feature embedding. Therefore, most existing 3D models do not pay much attention to this part, and trust the capabilities of network. Recently, Crasto et al. [22] trained a student network using RGB-frame input by learning feature representation from a teacher network, which had been trained using optical flow data to enhance temporal modeling. Similar solutions can be found in hidden two-stream networks [70] and D3D [71], which also used the other training step to enhance temporal modeling. Zhang et al. [66] used a 4D convolution to ensemble information from 3D ConvNets, which makes the model more complicated.

Our proposed two-path method consists of an appearance path using a 2D ConvNet only to extract appearance features and a motion path using 3D ConvNet to calculate motion features. Temporal modeling only exists in the motion path. The use of residual clips differs from that of RGB video clips because for residual clips, motions exist not only in the temporal dimension of residual frames, but also in the spatial dimension because one residual frame is generated from two adjacent frames.

Besides, we also want to mention that recently, with the application of transformers [72] in computer vision [73, 74], transformer based methods have also been used in video understanding [75–79]. The temporal modeling part is conducted by the self-attention mechanism across video patches. However, currently, the computational complexity of this kind of technology is very high for both training and inference part.

2.2.3 Two-Stream Modeling

Two-stream models usually stand for those methods combining 2D features / results from RGB stream with optical flow stream [21, 53, 54]. Some researchers extended the concept by combining the RGB-frame-input path with another path which uses pre-computed extra motion features, such as trajectories [80] or SIFT-3D [81], as well as optical flow. Then, many existing methods can be extended by combining motion feature stream to further improve their performances [7, 9, 22]. To distinguish our proposal from the aforementioned two-stream methods, we refer to our multi-

branch solution as '**two-path**' rather than 'two-stream' because we do not use any pre-computed motion features.

2.3 Proposed Method

In this section, we first introduce our proposal that uses residual frames as a new form of input data for 3D ConvNets. Because residual frames lack enough information for objects, which are necessary for the compound phrases used for label definitions in some video recognition datasets, we further propose a two-path solution to utilize appearance features as an effective complement for motion features learned from the residual inputs.

2.3.1 Residual Frames with 3D ConvNets

For 3D ConvNets, stacked frames are set as input, and the input shape for one batch data is $T \times H \times W \times C$, where T frames are stacked together with height H and width W, and the channel number C is 3 for RGB images. We denote the data as THW for simplicity. The convolution kernel for each 3D convolutional layer is also in three dimensions, being $k_T \times k_H \times k_W$. Then for each 3D convolutional layer, data will be computed among three dimensions simultaneously. However, this is based on a strong assumption that motion features and spatial features can be learned perfectly at the same time. To improve performance, many existing 3D models expand weights from 2D ConvNets to initialize 3D ConvNets, and this has been proved to provide higher accuracies. Pre-training on larger datasets will also enhance performance when fine-tuned on small datasets.

When subtracting adjacent frames to get a residual frame, only the frame differences are kept. In a single residual frame, movements exist in the spatial axis. Using residual frames for 2D ConvNets have been attempted and proved to be somewhat effective [55]. However, because actions or activities are complex with much longer durations, stacked frames are still necessary. In stacked residual frames, the movement does not only exist in the spatial axis, but also in the temporal axis, which is more suitable for 3D ConvNets because 3D convolution kernels will process data in both spatial and temporal axes. Using stacked residual frames helps 3D convolution kernel to concentrate on capturing motion features because the network does not need to consider the appearance information of objects or backgrounds in videos.

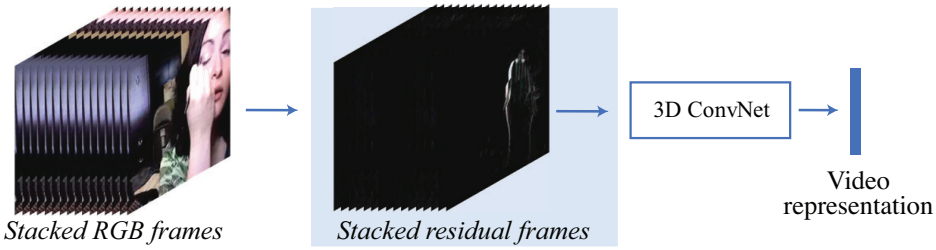


Figure 2.2: Residual frames with 3D ConvNets, which can extract better temporal representations.

Here we use $frame_i$ to represent the i_{th} frame data, and $Frame_{i\sim j}$ denotes the stacked frames from the i_{th} frame to the j_{th} frame. The process to get residual frames can be formulated as follows,

$$ResFrame_{i\sim j} = |Frame_{i\sim j} - Frame_{i+1\sim j+1}|. \tag{2.1}$$

The computation cost[1] is cheap and can even be ignored when compared with the network itself or optical flow calculation. For the computation cost of optical flow, It takes about 48 seconds for a 6-second video (165 frames) using the TV-L1 optical flow algorithm [82] and OpenCV on CPU. Though it can be accelerated by parallel computing, it is still time consuming compared to the inference time of our motion path (less than 0.19 seconds/video).

With this simple change (Fig. 2.2), 3D ConvNet can extract motion features by focusing on the movements in videos alone. However, by ignoring objects and backgrounds, some movements in similar actions become indistinguishable. For example, in the actions *Apply Eye Makeup* and *Apply Lipstick*, the main difference lies in the location of the movement being around the eyes or the mouth rather than the movement itself. In this example, 3D ConvNets may be able to distinguish them to some extent but the loss of information does increase the difficulty. Therefore, we further use a 2D ConvNet to process the lost appearance information and combine it with a 3D ConvNet using residual frames as input to form a two-path network.

2.3.2 Two-Path Network

Our two-path network is formed by a motion path and an appearance path, which is illustrated in Fig. 2.3.

[1]Considering the input data shape is $16\times112\times112$ in *THW* format, the cost to generate the residual clip is only about 0.6 MFLOPs, while it is usually larger than 10 GFLOPs for 3D ConvNets.

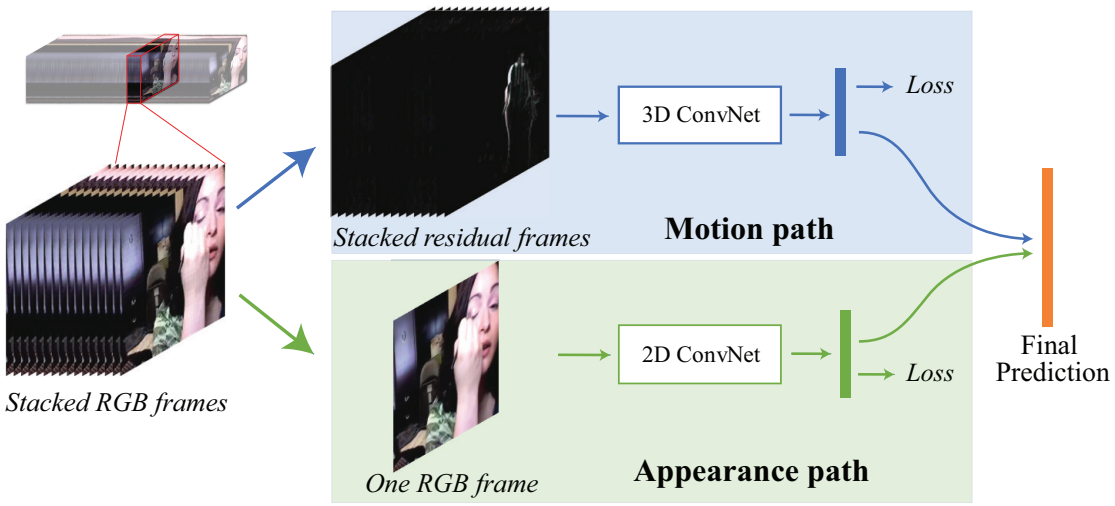


Figure 2.3: Framework of our two-path network. The motion path and the appearance path are trained separately using cross-entropy loss. Action recognition is carried out within each path. In inference period, the output probabilities from two paths are averaged. In this way, both motion features and appearance features are utilized for final classification. Note that the key for our motion path and appearance path is the input modality, any kinds of network architectures are potential options.

Motion path. Because residual frames are used in this path, movements then exist in both the spatial axis and the temporal axis. Therefore, 3D convolution layers are used in this path. Because there are many existing 3D convolution based network architectures which have been proved effective in many action recognition datasets, we do not focus on designing a new network architecture. Instead, to verify the robustness and versatility of our proposal, we conduct experiments on various models, and discuss especially on ResNet-18-3D for its good performance, which is a 3D version of ResNet-18 [60].

Appearance path. By using residual frames with 3D ConvNets, motion features can be better extracted, while foreground object appearances are almost lost. The lost part can be extracted by a 2D ConvNet, which uses one RGB frame as input. The goal for our appearance path is to embed object and background appearances which are mostly lost in the motion path. Therefore, to some extent, in contrast to TSN [48] or other complex models, a simpler 2D ConvNet is sufficient. The naive 2D ConvNet treats action recognition as a simple image classification problem. During training, only one frame in a video is randomly selected in one epoch.

Path fusion strategy. For the combination of these two paths, we average the predictions for the same video sample. We have experimented the late fusion

Table 2.1: Exact numbers used in datasets. * indicates the statistical information of the first split, and for other splits, the numbers are similar.

Dataset	UCF101*	HMDB51*	Mini-Kinetics	Kinetics	Sth-sth-v1	Sth-sth-v2
Classes	101	51	200	400	174	174
Train	9,537	3,570	74,262	225,231	86,017	159,742
Test	3,761	3,196	4,739	18,556	11,522	23,408

approach, which trained an additional classification layer to fuse features from different paths. The differences are limited and we discuss it in Sec. 2.7.2. There are other fusion methods that may be more effective, which we leave as our future work. We also want to address that this simple appearance path is a complement of the motion path, which can be replaced by any existing 2D solution (e.g., TSN [48]) to enhance the performance while increasing the complexity.

2.4 Experiments

2.4.1 Datasets and Metrics

Datasets. There are several commonly used datasets for video recognition tasks. Thanks to the large number of videos and labels in these datasets, deep learning methods can detect a large amount of actions. We mainly focus on the following benchmarks: UCF101 [56], HMDB51 [57], Something-something (Sth-sth) v1 and v2 [58], and Kinetics400 [45].

UCF101 consists of 13,320 videos in 101 action categories. HMDB51 is comprised of 7,000 videos with a total of 51 action classes. The Sth-sth datasets contain more than 100,000 videos across 174 action classes, which are highly related to human-object interaction and the definition of different actions is about the movement itself such as *Holding [something]*, where the object to be held has many options. The total video numbers are 108,499 for Sth-sth-v1 and 220,847 for Sth-sth-v2. Kinetics400 consists 400 action classes and contains around 240k videos for training, 20k videos for validation and 40k videos for testing. For the Kinetics400 dataset, because it is very large, we mainly perform our toy experiments on its subset, Mini-Kinetics [10], which consists of 200 action classes with 80,000 videos for training and 5,000 videos for validation.

The actual data used in our experiments may be a little smaller because some videos were unavailable from the corresponding websites. And for some datasets,

labels for testing dataset are unavailable. Therefore, following a standard procedure, we use the validation dataset to evaluate in such cases. The exact video numbers used for each datasets in our experiments are demonstrated in Table 2.1.
Metrics. We report top-1 and top-5 accuracies for all experiments for video action recognition. The performance on Mini-Kinetics, Kinetics, and Sth-sth (v1 and v2) is evaluated on the validation split. We also use correlation coefficient indexes of per-category accuracy for deeper analysis between different models, which may indicate the relationships between the knowledge learned from existing models. When testing the generalization ability using video retrieval task, k-nearest-neighbor (kNN) search is used and top-1, top-5, top-10, top-20, and top-50 performance will be reported.

2.4.2 Training from Scratch and Fine-tuning

There are always two ways to train a network, either training from scratch or fine-tuning from a pre-trained one. There is an obvious gap between these two training routes. Thanks to the proposal of the Kinetics datasets, several 3D convolution based models have been proposed with better performances using pre-trained models. Therefore, many recent works based their results on fine-tuned models for small datasets such as HMDB51 and UCF101, and trained from scratch for larger datasets such as Kinetics400 and its subset (i.e., Mini-Kinetics), as well as Sth-sth datasets.

Models can benefit from larger datasets, but training on larger datasets significantly increases computation time. For example, the size of Kinetics-400 dataset is almost 26 times of UCF101 dataset. The number of videos in the recent Kinetics-700 [46] dataset is around 50 times of UCF101. If training on UCF101 requires 1 day on one GPU, it will take nearly two months for the same experimental devices on Kinetics-700. Although improvements can be achieved, repeatedly increasing the size of datasets to improve performance is not always a solution. Therefore, in this work, in addition to the default settings discussed above, we also look into the situation that no additional knowledge is available. Specifically, we want to explore the limitations for 3D ConvNets on UCF101 and HMDB51 without any additional knowledge from other datasets.

2.4.3 Implementation Details

Motion path. In this path, stacked residual frames are set as the network input data. Residual frames are used identically to traditional RGB frame clips. For 3D ConvNets

in action recognition, there are several input setting choices. 3D ConvNets started from [6] which used a clip of 16 consecutive frames, with a 112×112 slice cropped in the spatial axis. To achieve the state-of-the-art results, clips in size $64 \times 224 \times 224$ were used in many recent works [10, 11]. When using such a large input data size, improvements can be always achieved while the training time are almost 16 times as long as before, as well as much larger memory occupations. Therefore, if not specified, for all of our motion path, following [6], frames are resized to 171×128 and 16 consecutive frames are stacked to form one clip. Then, random spatial cropping is conducted to generate an input data of size $16 \times 112 \times 112$. Before it is fed into the network, random horizontal flipping is performed. Jittering along the temporal axis is applied during training. The backbone in most of our experiments is ResNet-18-3D. R(2+1)D, I3D, and S3D are also tested to verify the robustness of our proposal. The batch size is set to 32. When models are trained from scratch, the initial learning rate is set to 0.1. We trained models for 100 epochs on UCF101, HMDB51, and Sth-sth (v1 and v2) datasets, and used 200 epochs for Mini-Kinetics. When fine-tuning on UCF101 and HMDB51 using Kinetics400 pre-trained models, model weights are taken from [8] and the network architecture remains the same. The initial learning rate is set to 0.001, and 50 epochs are sufficient.

Appearance path. In contrast to TSN [48], our appearance path uses a simpler model which treats action recognition as image classification because appearances in consecutive frames change infrequently, and the goal for this path is to capture appearance features for background and objects. Frames are first resized to 256×256. Then random spatial cropping and random horizontal flipping are applied in sequence to generate input data with a size of 224×224. This progress is standard in image classification to enable the use of many pre-trained models. ResNet-18, ResNet-34, ResNet-50, and ResNeXt-101 are used to test the impact of different model depth. In addition, models are also trained from scratch to see the performances when no additional knowledge is provided.

Testing and results accumulation. There are two means of testing for action recognition using 3D ConvNets. One is to uniformly get video clips from one video, which means a fixed number of clips is generated and set as the input of the model, regardless of the video length. The predictions are averaged over all video clips to generate the final result. The other method uses non-overlapping video clips, which means longer videos will produce more video clips. The final result for one video is also generated by averaging these video clips. We performed a small test for these two means of testing and found the difference can be ignored because all of

Table 2.2: Results on the UCF101 *split* 1, all models are trained from **scratch**. All models here can be treated as the motion path.

Model	**residual**	Top-1	Top-5
ResNet-18	×	61.6	84.9
ResNet-18	✓	**78.0**	**94.0**
R(2+1)D [7]	×	51.8	79.2
R(2+1)D [7]	✓	**66.7**	**88.3**
I3D [9]	×	56.5	81.3
I3D [9]	✓	**66.6**	**87.0**
S3D [10]	×	51.1	77.4
S3D [10]	✓	**64.8**	**86.9**

the clip results are averaged in both methods. Thus, we use the uniform method in our experiments, and our appearance path uses a fixed number frames sampled from all video frames to match the motion path.

2.5 Results and Analysis

In this section, results from single paths are introduced first. The motion path is used to investigate the effectiveness of stacked residual frames. Second, results from the appearance path are reported. Further analysis is conducted to explore the connections between models, especially the RGB 2D model and the RGB / residual 3D model. Finally, we show the performance of our proposed two-path network comparing to various existing models.

2.5.1 Single Path

Motion Path

Compared to RGB clips, stacked residual frames maintain movements in both spatial and temporal axes, which takes greater advantage of 3D convolution. Results are shown in Table 2.2 and the following discussion is all based on this table. By simply replacing RGB clips with our proposed residual clips, ResNet-18-3D results can be improved from 61.6% to 78.0%. To the best of our knowledge, this outperforms the

Table 2.3: Top-1 results for motion path on three benchmark datasets. Only motion path is tested. Pre-trained models are from RGB modality trained on Kinetics-400.

Model	Type	Pre-train	UCF101	HMDB51	Mini-Kinetics
ResNet-18	RGB	×	61.6	22.2	**65.0**
ResNet-18	Residual	×	**78.0**	**34.7**	64.4
ResNet-18	RGB	✓	84.4	**56.4**	-
ResNet-18	Residual	✓	**89.0**	54.7	-

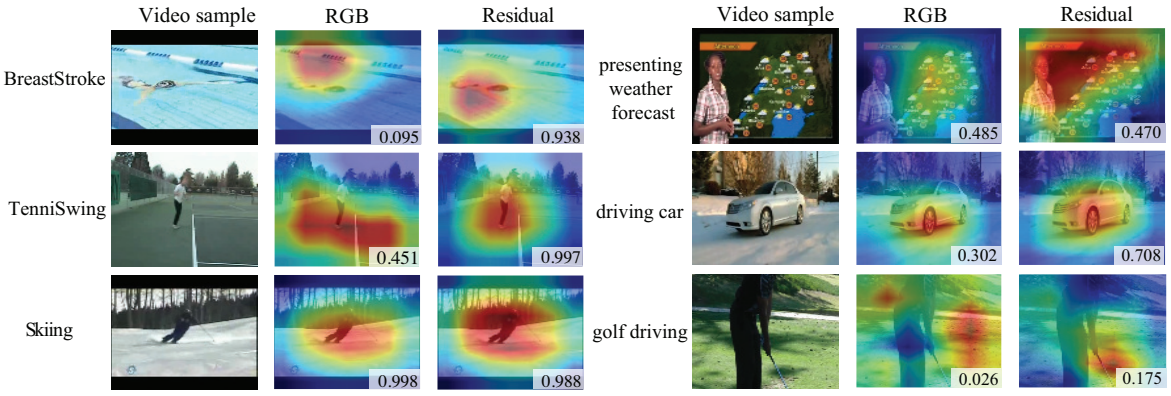


Figure 2.4: Visualization of motion path models using grad-cam [1]. The number is the corresponding prediction probability for each sample. Residual-input model focused more on the moving entity and the moving area while RGB-input included more background information.

current state-of-the-art results when models are trained from **scratch** on UCF101. R(2+1)D, I3D, and S3D are also experimented and improvements are achieved by more than 10% points when replacing original RGB input with our residual frames. Considering the training conditions that UCF101 is a small dataset and models can obtain almost 100% accuracy on the training split, the high performance of our residual input model indicate that residual data have better generalization ability than RGB counterparts for 3D ConvNets.

To sum up our residual inputs, we can see that this approach is robust for different model architectures. Because ResNet-18-3D is light-weighted and has good performance, we used ResNet-18-3D as the default backbone in our motion path.

We also tested the performance on HMDB51 and Mini-Kinetics. Results are shown in Table 2.3. On the HMDB51 *split* 1, the results can be improved from 22.2%

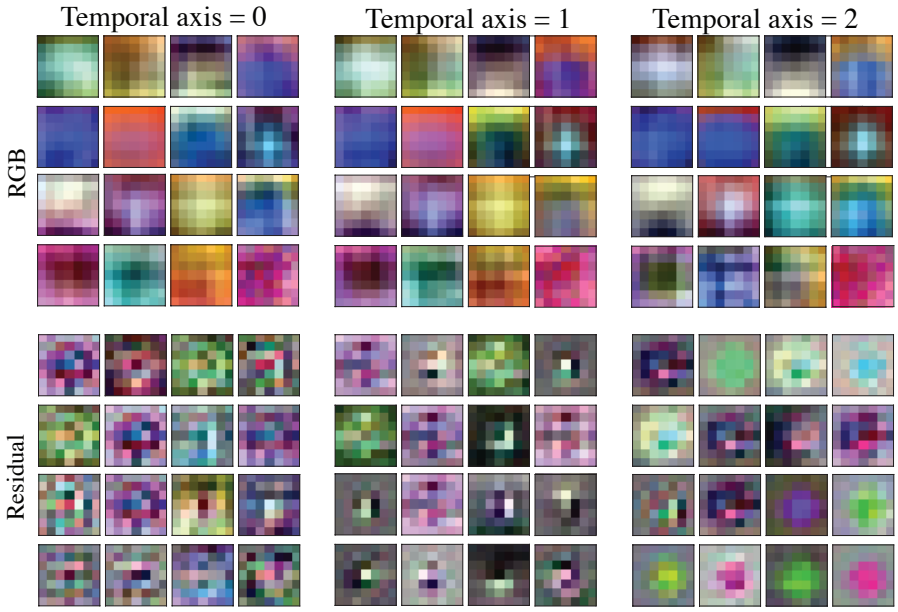


Figure 2.5: Visualization of model weights for the motion path. Models are trained from **scratch** on the Mini-Kinetics. Different model weights along the temporal axes are in charge of aggregating information from different temporal positions. If 3D convolution kernel weights are the same along the temporal axis, the convolution process equals to that using a 2D convolution kernels to process each channel separately, where there is no need to use 3D ConvNets. Filters in the RGB-input model are similar among temporal axis. On the other hand, in the residual-input model, the weights indicate that the residual-input model is more sensitive for changes in temporal dimension.

to 34.7% when replacing the original input with residual frames. However, the improvement cannot be observed for Mini-Kinetics because the labels are more related to objects rather than actions, which is also the main reason of introducing our appearance path. We would like to clarify again that the results in the table are from our motion path only. Residual-input model can also benefit from pre-trained models when fine-tuning, yielding 89.0% on the UCF101 *split* 1. The results on HMDB51 are not as good as the RGB model because on this dataset, the range of one variation of one action is larger. For example, the category *Dive* including bungee jumping and a movement by a score keeper on the ground. And many movements are inconsistent in one category while the samples are few, which greatly increases the difficulty for residual inputs.

For deeper analysis, we further use grad-cam [1] for visualization. As shown in Fig. 2.4, the residual-input model pays attention to the action entity while the

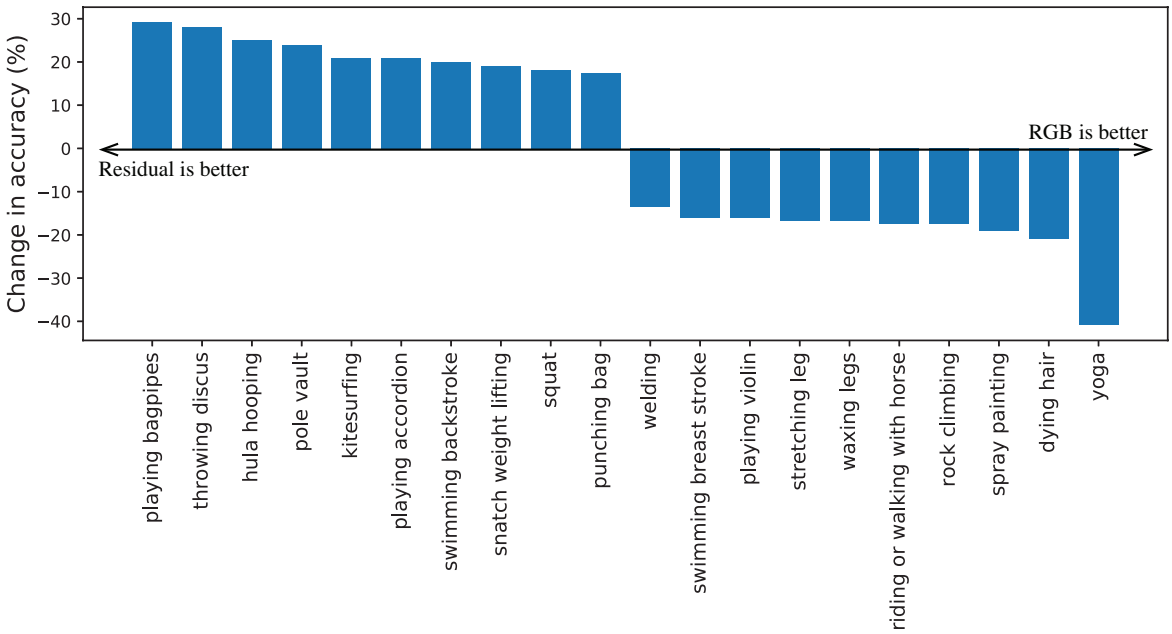


Figure 2.6: Accuracy difference between models with residual inputs and RGB inputs on Mini-Kinetics. Best-10 and worst-10 categories are illustrated.

RGB-input model focuses more on the background. The prediction probability is low for *BreastStroke* because RGB model gives higher probability for another swimming style *FrontCrawl*.

The first 16 out of 64 convolutional filters in the *conv*1 layer from the RGB-input model and the residual-input model are illustrated in Fig. 2.5. These two models are both trained from scratch on Mini-Kinetics, with the same hyper-parameters in settings. We can see that the filters in the RGB-input model are similar among different temporal axes. Notice that these similar weights are from models with random initialization. If 3D ConvNets are initialized by duplicating 2D ConvNet weights pre-trained on ImageNet [83], this phenomenon will also happen because the model weights are born to be similar across the temporal axis, although better results can be achieved in such a condition. The filters in residual input model differs from each other among different temporal axis, indicating that this model is more sensitive to the changes in time. The accuracy differences between our residual-input model and the RGB-input model are illustrated in Fig. 2.6. We show the best-10 and worst-10 classes. The positive peak belongs to the class *playing bagpipes* and we find that in this category, there are global movements caused by lens shake and other irrelevant movements by bystanders, which can be handled by our residual-input model. We think it is because with residual inputs, the global movement or such

Figure 2.7: Case study. Bette performance can be achieved using residual clips in category **Throwing discus** while in category **Yoga**, RGB clips perform better. In the presented samples, it is obvious that common movements (e.g., turning around) exist in different samples in category **Throwing discus**. However, the definition **Yoga** is vague and the style of clothes may play a more important role than the movements.

shaking noise has been considered during training, making the model robust for these cases. We also illustrate case studies in Fig. 2.7. It is clear that movements in *throwing discus* are highly consistent. In contrast, movements in *yoga* varies from each other while the appearance information plays a more important role.

Based on our analysis, the ability of 3D ConvNets with traditional RGB video clips as input may be limited because of the ambiguity in action labels. Additionally, more attention is paid to appearance rather than movements for RGB 3D models because during training, the network finds that appearance features are discriminative enough in such datasets.

The aforementioned results and analyses are on the UCF101, HMDB51, and Mini-Kinetics datasets. We also conduct paired experiments on the Kinetics400 and Sth-sth (v1 and v2) datasets with RGB and our residual clips. Results are in Table 2.4. We also list references to validate that our experimental settings are close to standard baselines even without carefully choosing training hyper-parameters. Experiments in each block share **exactly the same** hyper parameters for training. Specifically, scratch training took 100 epochs and fine-tuning took 60 epochs. State-of-the-art

Table 2.4: Results on Kinetics400 and Something-something datasets (v1 and v2). Our experiments in the same block used exactly the same settings.

	Pre-train	Modality	Input size	Top1	Top5
Kinetics400					
ResNet-18 [8]	-	RGB	16×112^2	54.2	-
ResNet-18 [7]	-	RGB	16×112^2	64.2	-
ResNet-18	-	RGB	16×112^2	**62.3**	**84.3**
ResNet-18	-	Res	16×112^2	60.5	82.7
ResNet-50 [84]	-	RGB	32×112^2	64.3	-
ResNet-50	-	RGB	8×224^2	54.0	78.6
ResNet-50	-	Res	8×224^2	**56.1**	**79.4**
Sth-sth-v1					
ResNet-18 [85]	K400	RGB	32×112^2	43.1	-
ResNet-18	K400	RGB	16×112^2	29.8	58.9
ResNet-18	K400	Res	16×112^2	**30.5**	**60.0**
ResNet-18	-	RGB	16×112^2	32.0	59.1
ResNet-18	-	Res	16×112^2	**35.2**	**63.9**
ResNet-18	K400	RGB	32×112^2	42.2	70.9
ResNet-18	K400	Res	32×112^2	**44.2**	**74.5**
Sth-sth-v2					
ResNet-50 [84]	-	RGB	32×112^2	44.3	-
ResNet-18	K400	RGB	16×112^2	44.9	74.3
ResNet-18	K400	Res	16×112^2	**46.2**	**75.8**
ResNet-18	-	RGB	16×112^2	46.9	76.4
ResNet-18	-	Res	16×112^2	**47.0**	**77.9**
ResNet-50	-	RGB	8×224^2	36.9	67.2
ResNet-50	-	Res	8×224^2	**39.5**	**70.3**

methods usually used longer epochs (e.g., 200 or longer epochs for ResNet-50-3D network) to train, while in this table, we focus more on the effectiveness of data modality.

From Table 2.4, we can find that on Kinetics400 dataset, RGB modality performs better than our residual clips when using ResNet-18-3D network, which is reasonable and consistent with our findings from the Mini-Kinetics dataset. However, when using ResNet-50-3D, our residual-input model outperforms the RGB counterpart. We think this is because networks are trained for only 100 epochs, which are enough to reflect that with our proposed residual input, motion features can be easier learned than using RGB video clips. With the same network architecture, the epoch number is 256 in [11] and 250 in [84].

Table 2.5: Action recognition accuracies different datasets. The input is only one single frame, which can be seen that no temporal information is used here even for action recognition.

Dataset	UCF101		HMDB51		Mini-Kinetics	
Pre-train	Scratch	ImNet	Scratch	ImNet	Scratch	ImNet
ResNet-18	37.7	79.6	25.0	42.6	57.7	64.4
ResNet-34	40.1	81.5	24.8	43.1	59.4	68.9
ResNet-50	33.7	83.8	21.3	43.4	58.6	69.7
ResNeXt-101	34.4	85.2	23.3	45.6	59.7	70.5

Compared to the Kinetics400 dataset, the Sth-sth-v1 and Sth-sth-v2 datasets are more temporal related datasets. On the Sth-sth datasets, with our proposed residual clips, better performance can be achieved across different settings such as different parameter initialization strategies (i.e., scratch training and fine-tuning), different input sizes (i.e., $8\times224\times224$, $16\times112\times112$, and $32\times112\times112$), and different network architectures (i.e., ResNet-18-3D and ResNet-50-3D). Through our analyses, our residual-input model can capture more temporal related features. The better performance of residual input in this table further proves our statement because the Sth-sth datasets are more temporal related datasets while Kinetics-400 dataset cares more about the appearance (scene/object) information. Another interesting finding is that on the Sth-sth datasets, better performance can be achieved when models are trained from scratch than those fine-tuned from Kinetics400 pre-trained weights, which reveals that the main useful features for action recognition on these datasets are different.

Appearance Path

For the appearance path, four ResNet architectures were used, namely ResNet-18, ResNet-34, ResNet-50, and ResNeXt-101. Scratch training as well as fine-tuning from ImageNet pre-trained models were both tried. The results are shown in Table 2.5.

We can clearly find that the gap is large for 2D ConvNets between these two training ways, which is consistent with previous works on image classification tasks. However, pre-training also needs much time if no pre-trained models are provided. For better performance, deeper networks generally provide higher scores.

Regarding Mini-Kinetics, ImageNet pre-trained models were directly used and high accuracies could be achieved. Among these 2D ConvNets, the best top-1 accuracy was 70.5%, which is very high. However, in this case, the action recognition task is treated as a simple image classification task, which does not benefit from the use of any temporal information.

The performance of ResNet-18-2D using pre-trained weights is 79.6% in UCF101 dataset, which is close to the performance of scratch training using ResNet-18-3D in Table 2.2, 78.0%, though it may be unfair to compare these two models because the 2D version utilizes image classification knowledge to initialize its parameters while the 3D version does not. 3D convolution based models are thought to have better ability for the extraction of spatio-temporal features than 2D convolution, the results here indicate that spatial information may be sufficient for many cases in some video recognition datasets. Pre-training 3D ConvNets on complex video datasets could be a good solution. However, it is still prone to mainly using appearance features, which is actually not in line with the original intention for video representation because temporal information is lost.

Analysis Among Models

The difference between 2D convolution and 3D convolution is that 3D convolution has another dimension which is aimed to process temporal information. For continuous frames, especially those trimmed videos provided in video recognition datasets, the difference between frames is limited. Therefore, the 3D convolution may not process temporal information efficiently. Duplicating ImageNet pre-trained model parameters as the initial model parameters does provide improvements, but spatial-temporal convolution might be lazy during fine-tuning process because even for models trained from scratch, model weights tend to be similar among temporal axes (Fig. 2.5).

Here, we introduce the correlation coefficient index to calculate the relationships between different models. 2D models and 3D models were tested. We also used optical flow streams with both 2D ConvNet and 3D ConvNet as comparative models. Correlation coefficient indexes for per-category accuracies between two different models are reported in Table 2.6. The backbone networks are ResNeXt-101-2D and ResNet-18-3D for 2D ConvNet and 3D ConvNet, respectively. All models were fine-tuned from pre-trained ones to ensure the classification performance. From the table, we can see that the correlation coefficient index for the RGB 2D and 3D models is high (Tag A), which indicates that these two approaches may make

Table 2.6: Correlation coefficient indexes for per-category accuracy on the UCF101 *split* 1. *Type* means the type of convolution kernels used in the network.

Tag	Model		Model		Correlation
	Input	Type	Input	Type	
A	RGB	2D	RGB	3D	**0.839**
B	RGB	2D	Residual	3D	**0.663**
C	RGB	2D	Flow	2D	0.505
D	Flow	2D	RGB	3D	0.569
E	Flow	2D	Residual	3D	0.534
F	RGB	3D	Residual	3D	0.791
G	Flow	2D	Flow	3D	0.582
H	Flow	3D	RGB	2D	0.478
I	Flow	3D	RGB	3D	0.612
J	Flow	3D	Res	3D	**0.742**

judgement in a similar way while optical flow stream differs significantly. Though our residual-input model has high correlation with the RGB 3D models (Tag F), the correlation becomes lower with the RGB 2D models (Tag B) because using residual frames results in more motions being used for classification rather than appearance. From Tag G, an interesting finding is that even both use optical flow data, the 2D model and the 3D model conduct judgement differently, which might be caused by the training ways because 3D ConvNets use stacked optical frames while our 2D ConvNets are too naive and the prediction is from one single time spot. From Tag H and Tag I, we can see the correlations between the RGB model and optical flow models are not high. For the optical flow 3D model, the highest correlation comes from a comparison with the residual 3D model, indicating that **the behavior for our motion path is similar to optical flow.**

2.5.2 Two-Path Network

By combining the motion path with the appearance path, appearances and motions can be used to get the predictions. Because we have several models, we tried different combinations among different models. For example, in the UCF101 dataset, we tried different combinations by selecting two models among RGB 2D model, optical flow 2D model, RGB 3D model, optical flow 3D model, and residual 3D model. The results are listed in Table 2.7. In our implementation, the optical flow path used a

Table 2.7: Results from different combination of different models on the UCF101 *split* 1. Our combination yielded the best performances.

Appearance path		Motion path		Top-1	Top-5
Input	Type	Input	Type		
RGB	3D	Optical flow	2D	75.7	92.1
RGB	3D	Optical flow	3D	87.6	97.3
RGB	3D	Residual	3D	87.4	97.5
RGB	2D	Optical flow	2D	75.7	92.1
RGB	2D	RGB	3D	86.6	97.1
RGB	2D	Optical flow	3D	**90.3**	**98.5**
RGB	2D	Residual	3D	**90.3**	**98.5**

ResNeXt-101 backbone, which is the same as our appearance path. However, the combination of 2D optical flow model and other RGB models produces side effects on the accuracies. When the 2D RGB model is set as the appearance path, the results are the same when setting the 3D optical flow model or the 3D residual model as the motion path. We think this is another proof of the effectiveness of our residual-input model for motion feature extraction.

2.5.3 Comparison with Other Methods

We do not focus on developing a new network architecture. Therefore, we only compare our method with some corresponding methods, as shown in Table 2.8. Our single motion path can outperform TSN (RGB or RGB difference) [48] and I3D-RGB [9] which only use RGB input data. TSN also tried a multi-path solution, which combined RGB modality with RGB difference and it only achieved 87.3% on UCF101 while our single motion path can obtain 89.0%. Without any additional computation for optical flow and only using ResNet-18-3D, we can even have better performance than the original two-stream model [54] which uses optical flow. On the other hand, our model is not better than the state-of-the-art such as [9]. But it is out of the scope of our proposal because many settings including the input size and network architectures are totally different.

We also show the computation complexity in this table. The GFLOPs in Table 2.8 only represents the complexity of the networks themselves, and these methods using the optical flow stream requires much additional computation which cannot be

Table 2.8: Comparisons on UCF101, HMDB51 and Kinetics400. * indicates methods using optical flow. The computational complexity for optical flow is not included.

	GFLOPs	UCF101	HMDB51	Kinetics400
Scratch training				
ResNet-18-3D baseline [8]	16.8	42.4	17.1	54.2
STC-ResNet-101 [86]	15.6	45.6	-	64.1
NAS [64]	-	58.6	-	-
TSN (RGB only) [48]	4.1	48.7	-	-
C3D [6]	38.5	51.6	24.3	55.6
ResNet-18-3D (residual clips, ours)	16.8	**78.0**	**43.7**	-
Single path (fine-tuning)				
CoViAR (Residuals) [55]	4.2	79.9	44.6	-
TSN (RGB difference) [48]	4.1	83.8	-	-
TSN (RGB) [48]	4.1	84.5	-	-
ResNet-18-3D baseline [8]	16.8	84.4	56.4	-
C3D [6]	38.5	82.3	51.6	-
I3D (RGB, ImNet pre-train) [9]	107.8	84.5	49.8	71.1
R(2+1)D (RGB)	152.4	96.8	74.5	74.3
ResNet-18-3D (residual clips, ours)	16.8	**89.0**	**58.1**	62.9
Multi-path (fine-tuning)				
Two-stream* [54]	3.3 +	86.9	58.0	65.6
Two-stream (+SVM)* [54]	3.3 +	88.0	59.0	-
TSN (RGB + RGB diff) [48]	8.2	87.3	-	-
I3D* [9]	215.6 +	**98.0**	**80.7**	**74.2**
CoViAR (3 nets) [55]	12.6	90.4	59.1	-
Two-path (ours)	32.8	**90.6**	56.6	67.7

ignored. Though single R(2+1)D can achieve good performance, the price (152.4 GFLOPs) is around 9 times larger than ours when compared with our single residual input path. For the extraction of optical flow, according to [87], the GFLOPs for three well-known network, FlowNet [88], FlowNet2 [89], and PWC-Net+ [90], are 66.9, 368.3, and 101.6, respectively. Even the computational complexity is only 3.3 GFLOPs for the two-stream model [54], it will increase by at least 22 times and surpass our two-path solution when taking the calculation of optical flow into consideration.

For the Mini-Kinetics dataset, results are shown in Table 2.9. We mainly compared our method with TBN [49] and MARS [22], which does not use optical flow yet achieving good performances. TBN used temporal bilinear modeling to process temporal information, which is insufficient to extract motion features compared with ours. The backbone network for MARS is ResNeXt-101-3D. To get the results using

Table 2.9: Results on Mini-Kinetics. Our tow-path network outperforms MARS even when it uses three streams. The depth of our motion path is 18 while that for MARS is 101.

Method	Optical flow	Top-1	Top-5
TBN C2D [49]	×	69.0	89.8
TBN C3D [49]	×	67.2	88.3
MARS [22]	×	72.3	-
MARS + RGB [22]	×	72.8	-
MARS + RGB + Flow [22]	✓	73.5	-
Motion path	×	64.4	86.4
Our two-path	×	**73.9**	**91.4**

distillation methods, their networks should be trained on optical flow inputs first, and then another network is built to learn features from optical flow stream. The process is complex and is much more expensive than our proposed two-path method. The backbone network for our motion path is ResNet-18-3D, which is shallower than that used in MARS. There is much room for our proposed solution to improve by using deeper networks and other feature fusion methods.

2.6 Generalization Abilities

In this section, we further analyze the generalization ability using residual inputs by two video understanding tasks. The first experiment uses trained models as a feature extractor and take video retrieval as a target task on unseen datasets. The second experiment sets residual clips as inputs and they are applied to existing self-supervised methods, where the baselines are all based on RGB inputs.

2.6.1 Video Retrieval on Unseen Datasets

When using a different dataset, if the trained model can capture video representations with sufficient information, better retrieval performance can be achieved because samples with the same label share similar movements. Here we trained two models on Kinetics400 using RGB clips and residual clips, respectively. The retrieval task is conducted on unseen datasets, UCF101, HMDB51, Sth-sth-v1 and Sth-sth-v2 [58]. The results are shown in Table 2.10.

Table 2.10: Results on video retrieval. Both RGB and residual models are trained on Kinetics400. The input size is $16 \times 112 \times 112$.

Tag	Modality	Train	Test	Top-1	Top-5
a	RGB	Kinetics400	UCF101	69.6	85.7
	Res	Kinetics400	UCF101	72.1	87.1
b	RGB	Kinetics400	HMDB51	31.5	61.5
	Res	Kinetics400	HMDB51	42.7	68.2
c	RGB	Kinetics400	Sth-sth-v1	4.2	13.5
	Res	Kinetics400	Sth-sth-v1	4.6	15.4
d	RGB	Kinetics400	Sth-sth-v2	5.3	17.1
	Res	Kinetics400	Sth-sth-v2	6.7	19.4

As we can see from the table, on the Kinetics400 dataset, the residual model can have better performance for video retrieval when applied to unseen datasets. We can obtain 2.5% points and 11.2% points improvements at top-1 retrieval accuracy on on UCF101 and HMDB51 (Table 2.10 a and b), respectively. the Sth-sth datasets contain more samples, which make it difficult to do retrieval, and spatiotemporal models usually do not have good performances even using supervised training. We still obtain better results on these datasets at both top-1 and top-5 accuracies (Table 2.10 c and d). We think this is an evidence that the residual model can learn high-quality video representation and has better generalization ability than the traditional RGB model. And training models with RGB video clips may be prone to overfitting.

2.6.2 Video Self-Supervised Learning

Self-supervised learning has drawn much attention recently because it does not require any labels while it can be utilized to train models to extract effective features. We adopted three recent works, 3DRotNet [4], CMC [2], and VCP [3], as our baselines. 3DRotNet trained a network by predicting the rotated degrees of input video clips. CMC used contrastive learning to constrain that features extracted from the same data should be similar, even though they are generated using different data augmentation strategies, or they belong to different color spaces such as RGB and Lab. VCP treated different transformations as labels and trained models to distinguish which transformation has been conducted before being fed into the network. We combine our residual input with their methods. For CMC, we treated RGB and residual

Table 2.11: Video retrieval performance on UCF101 and HMDB51 using self-supervised methods.

	Top-1	Top-5	Top-10	Top-20	Top-50
Dataset: UCF101					
3DRotNet [4]	14.2	25.2	33.5	43.7	59.5
3DRotNet [4] + res	**14.5**	**30.5**	**40.2**	**53.1**	**70.7**
VCP [3]	18.6	33.6	42.5	53.5	68.1
VCP [3] + res	**25.6**	**43.0**	**53.2**	**64.8**	**79.2**
CMC [2]	26.2	39.3	46.8	55.6	66.8
CMC [2] + res	**27.7**	**46.1**	**55.5**	**65.0**	**76.5**
Dataset: HMDB51					
3DRotNet [4]	**6.2**	18.7	31.0	46.6	70.5
3DRotNet [4] + res	6.0	**21.6**	**33.8**	**49.1**	**71.8**
VCP [3]	7.8	23.8	35.3	49.3	71.6
VCP [3] + res	**11.0**	**31.2**	**43.8**	**58.4**	**78.7**
CMC [2]	10.8	26.2	40.1	54.3	74.9
CMC [2] + res	**11.4**	**27.7**	**42.0**	**55.6**	**76.0**

inputs as two different views and for 3DRotNet and VCP, we simply replaced the RGB input as our residual ones. Therefore, we can say that all training settings remain the same except for the input data modality.

Models were only trained on the UCF101 *split* 1 and no labels were used. All these models used 3D convolutional networks. After training was done, we first treated the trained models as a feature extractor and conducted video retrieval on both UCF101 and HMDB51 datasets. The results are in Table 2.11. It is obvious that by using residual clips, better performance can be achieved. Though the models are not trained on HMDB51, we can still find improvements on video retrieval, which reveals that residual clips can help to train the model with better generalization ability.

These self-supervised models were also treated as a parameter initialization method. We fine-tuned these models on the first split of UCF101 and HMDB dataset to check whether it is also helpful for video recognition task. The results are shown in Table 2.12. Better performances can also be obtained by replacing RGB clips with our proposal.

For self-supervised learning methods, the network does not use any labels to train. Based on our findings, the network will be prone to capturing object appearance features by using RGB video clips with annotations in supervised learning. However,

Table 2.12: Comparison of action recognition accuracy on the UCF101 and HMDB51 *split* 1 using Self-supervised methods

Method	UCF101	HMDB51
3DRotNet [4]	62.9	33.7
3DRotNet [4] + res	**70.8**	**40.0**
VCP [3]	66.0	31.5
VCP [3] + res	**71.3**	**45.0**
CMC [2]	59.1	26.7
CMC [2] + res	**71.6**	**35.6**

Table 2.13: Comparisons between different sources of residual inputs using motion path. Results are reported on the UCF101 *split* 1

Model	Type	Pre-train	UCF101
ResNet-18	Gray	×	65.0
ResNet-18	Gray → Res	×	61.4
ResNet-18	RGB → Res	×	**78.0**
ResNet-18	Gray → Res	✓	87.8
ResNet-18	RGB → Res	✓	**89.0**

without label information, RGB models cannot recognize principle objects in videos such as "guitar" to get the "playing guitar" action category. And the advantage of our residual input will be amplified because the similarities of movements become the major clue. The success of our proposed residual clips in video self-supervised learning has also been obtained in some recent works [91, 92].

2.7 Discussions

In this section, we will pose further discussions on the advantage of our residual input model (i.e., motion path in our solution), and some additional results to hold our statements.

2.7.1 Residual Sources: Grayscale vs RGB

We conducted experiments using grayscale frames in three ways: 1. training from scratch; 2. generating residual frames and then training from scratch; 3. generating

Table 2.14: Comparison of different combination of two paths. Experiments are on the *split* 1 for the UCF101 and HMDB51 datasets.

Method	UCF101	HMDB51	Mini-Kinetics
Simple Comb.	90.3	**56.1**	**73.9**
Use fusion layer	**92.4**	50.3	72.7

residual frames and then using pre-trained models to fine-tune. Results are reported on the UCF101 *split* 1 in Table 2.13. For convenience, the input channel is still 3 which are duplicated from 1 grayscale channel. From the table, we find that when using residual clips, original RGB source frames are better. We infer that the three RGB channels capture motions in different dimensions. The more movement information input data contain, the better performance can be achieved.

2.7.2 Path Fusion Strategies

Experiments have been conducted over the three datasets (i.e., UCF101 *split* 1, HMDB51 *split* 1, and Mini-Kinetics). Results are reported in Table 2.14. For the UCF101 *split* 1, using an additional fusion layer is better while for HMDB51 datasets, simply combining prediction scores can obtain better performance. For Mini-Kinetics, though the gap is limited, simple combination can still yield 1.2% points advantage. Other fusion strategies might help, such as fusing mid-level features using by-pass connections as the SlowFast network [11]. However, different fusion strategies are not the main point of this work because the appearance path is to compensate for the lost of some spatial information in particular cases. And from current experimental results, we could only say that it highly depends on the datasets and usually a simple combination of prediction scores can already obtain good performance.

2.7.3 Comparison with Optical Flow Related Works

Optical flow is a useful modality for motion representations. If we only want to compare different modalities, regardless of the computational complexity, optical flow might be the best modality for training from scratch, achieving 81.2% [54] on the UCF101 *split* 1. However, to achieve better performance, almost all existing methods which used optical flow combined results with one RGB stream. For 3D ConvNets, the temporal stream in I3D [9] can achieve 85.8% accuracy using pre-trained weights, while with our residual inputs and ResNet-18-3D, we can obtain 89.0%. It is hard to

say which is better because of the setting differences. However, we can say that our residual input is another effective data modality.

There are also several works which have absorbed optical flow information using two-step training with frame reconstruction or knowledge distillation such as hidden two-stream network [70], MARS [22], and D3D [71]. To achieve state-of-the-art performance, in these papers, results were reported by using more frame numbers per clip, deeper network architectures, and ensemble models. It is hard to say which is better because of so many differences. A comparable settings advantage for our two-path solution is listed in Table 2.9. With the same input data sizes (i.e., 16 frames per clips and spatial resolution is 112×112), we can obtain 0.4% points improvement over MARS+RGB+Flow where MARS used three ResNeXt-101-3D models while we only use one ResNet-18-3D model for the motion path and one ResNeXt-101-2D model for the appearance path. In addition, our main focus is the novel data modality and its functional mechanism. There are much room to improve for our solution with deeper network architectures, larger inputs sizes, as well as using ensemble models.

2.7.4 Key Feature: Appearance vs Motion

To further explore the details, we conducted additional experiments with two subsets of Mini-Kinetics, where the dataset size is similar to UCF101. These two subsets are Mini-100 and Mini-200. Training and testing videos are all from Mini-Kinetics and for Mini-100, we use the first 100 categories with 100 videos per category to train. For Mini-200, we use a total of 200 categories with 50 videos per category to train. Results are in Table 2.15. All training hyper-parameters are the same. The initial learning rate is set to 0.1 with exponential decay. Apparently, residual-input model can outperform RGB-input model. Because there are very limited cases for each category, the consistent information among samples are motion features rather than appearance information, revealing that residual input can help 3D ConvNet focus on extracting motion features.

Results on Something-something datasets (Table 2.4) also imply that for large datasets, if the definition of actions is temporal related, motion clues play an important role, in which situation our residual-input model is more suitable and competent.

Table 2.15: Toy experiments on Mini-100 and Mini-200.

Dataset	Epochs	Modality	Top-1	Top-5
Mini-100	100	RGB	41.9	72.2
Mini-100	100	Res	45.8	76.2
Mini-200	100	RGB	28.0	55.4
Mini-200	100	Res	36.2	64.7

2.7.5 Potential Applicable Settings

In this section, we show basic studies on residual inputs with 3D ConvNets, without combining a lot of tricks. With the development of techniques in video understanding, different techniques and strategies have been proposed and can be combined with out proposal, which are not limited to 1) using larger data inputs (i.e., $64 \times 224 \times 224$); 2) using deeper network architectures (i.e., 101, 152 layers); 3) using stronger data augmentations; 4) using ensemble models; 5) combining with optical flow models; 6) using knowledge distillation; 7) designing new modules; 8) applying attention mechanism; 9) carefully setting hyper-parameters; and 10) using larger dataset (Sports-1M, IG-65M, etc.) to pre-train. Most state-of-the-art methods used several of these strategies. On the other hand, we would like to present a basic study of residual inputs and its mechanism rather than the accuracy numbers.

Based on our findings, it is very easy to combine our proposal with many existing works with promising performance. The great generalization ability also revealed that it might benefit from our residual inputs for many other video understanding tasks which directly use trained models as a feature extractor.

2.7.6 Limitations

One limitation for this solution is the computation complexity. Though we have built a more efficient model to extract better model representations without additional computation in optical flow, the network backbones are still 3D ConvNets. Compared to 2D convolution, the complexity of 3D convolution is high. Replacing some 3D convolutional layers with 2D ones and conducting temporal differences in feature spaces might be one possible direction. Another limitation is the lack of appearance information for the motion path. Though we propose a two-path solution to make up for the deficiencies for this part, it is hard to say this combination is a good way,

except for the better performance in numbers. Carefully designing a single network to embed both spatial and temporal information should be more efficient.

2.8 Conclusions

In this section, we mainly focused on extracting good video representations without optical flow in supervised learning paradigm. 3D ConvNets are believed to be capable of capturing motion features when RGB frames are set as input, but we demonstrated that it is not always true. We improved the use of 3D convolution by using stacked residual frames as the network input. The overhead for this computation was negligibly small. With residual frames, the results of 3D ConvNets could be improved significantly on benchmark datasets. With a simple appearance path compensating for the lost of some spatial information, the superior performance could be advanced and better or comparable results could be achieved compared with the corresponding two-stream methods. Extensive results and analysis imply that residual frames can be a fast but effective way for a network to capture motion features and they are a good choice for avoiding complex computation for optical flow. The applications to video retrieval on unseen dataset and video-based self-supervised methods show that residual input models can have better generalization abilities than the RGB counterparts.

Chapter 3

Inter-Intra Contrastive Learning for Self-Supervised Video Representation

3.1 Introduction

Video understanding tasks require good feature representations from videos. Tasks such as video segmentation, video summarization, and video retrieval rely on effective motion representation extractors, which are usually trained on the basis of video recognition. For video recognition, many works explore different network architectures [6–10, 48]. In addition to using RGB frames as input data, some other works try to utilize optical flow as an additional data modality to form a two-stream model for better motion feature extraction [21, 53, 54]. With optical flow, better results can be achieved [7, 9, 10]. Hara et al. [8] argued that video recognition can imitate image recognition procedures, which means that the performance can also be significantly improved with larger datasets.

To achieve better performance, video recognition datasets become larger and larger. And there are numerous unlabeled videos available on the Internet every day. Creating new video datasets with annotations requires a wealth of resources. In addition, video recognition tasks usually require properly trimmed action video clips to ensure the performance, which makes the situation more serious. Therefore, if unlabeled videos can be directly used to facilitate learning, numerous data could be utilized at no annotation cost. To address this issue, self-supervised learning is drawing more and more attention these days as it does not need any labels to train.

Self-supervised learning belongs to unsupervised learning. Many video-based self-supervised learning techniques originate from image tasks. Several dedicatedly

designed tasks such as solving jigsaw puzzles [93], image inpainting [94], and image color channel prediction [95] are proposed to learn image features. For video data, many existing works [5, 23–30] have been proposed to focus on temporal information such as making models sensitive to the temporal differences of input data. Most of the aforementioned methods can be classified into the pretext-task category. Here we call them as **intra-sample learning** because all operations are conducted within the sample itself. For example, transformations such as shuffling frames to change their orders or rotating video clips are conducted without using a different video sample.

In addition to intra-sample learning, contrastive learning is also an important branch of self-supervised learning techniques. Because contrastive learning [2, 31–43] tries to train the network by distinguishing one sample from another, we call this kind of methods **inter-sample learning** methods. Inter-sample learning techniques are also originated from images. Anchor and positive samples are different crops from the same image while negative samples are other crops from different images. If a model can distinguish whether these samples come from the same image or not, it is believed that the model can extract discriminative features from images. For video data, the procedure is almost the same, and the differences lie mainly in the input data and the feature extraction network.

For intra-sample learning methods, tasks should be carefully designed, whereas inter-sample learning methods focus more on the training strategies, other than the optimization tasks. Therefore, for inter-sample learning, whether good temporal information can be learned relies on the model itself. Furthermore, if spatial information in certain samples is sufficient enough, then temporal information would be ignored and the model will not be helpful if applied to other video tasks [96]. To address this issue, we try to force the model to capture rich temporal information. To do so, we introduce some transformations from intra-sample learning and apply them to the anchor sample. These transformations can break the temporal relationships of the video clip and then the intra-negative samples are generated. The models can learn richer temporal differences as well as spatial differences by contrastive learning with the anchor, intra-positive, inter-negative, and intra-negative samples. And we call this learning scheme **I**nter-**I**ntra **C**ontrastive (IIC) learning. We illustrate the general idea of our method in Fig. 3.1. Note that the constraint between the anchor and the intra-positive sample is that they come from the same instance, and different modalities can be utilized.

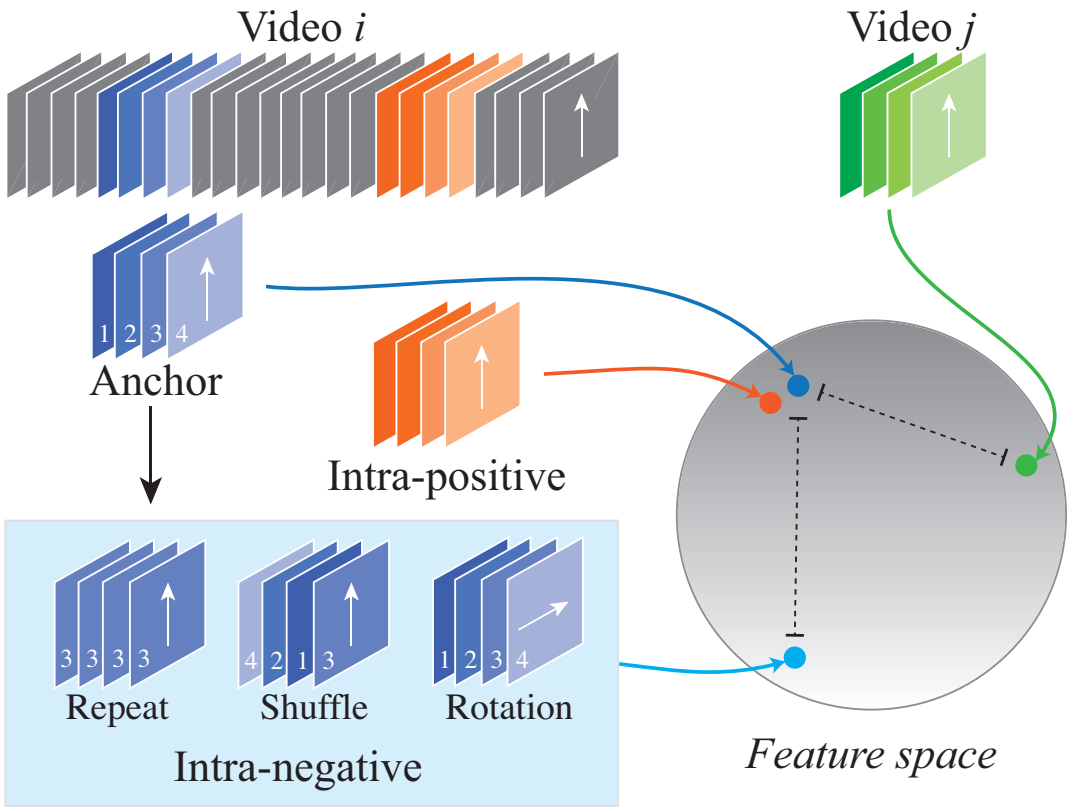


Figure 3.1: General idea of IIC. Given video i and video j, two sampled video clips from video i are treated as the anchor and intra-positive samples, whose features are constrained to be similar to each other. Data sampled from video j is treated as the negative sample. We generated intra-negative samples from the anchor sample by breaking its temporal relations, which can be treated as hard-negatives because they share similar spatial information but different motion features, and can force the model to learn better more discriminative temporal information.

To the best of our knowledge, we are the first to introduce intra-negative samples to videos in contrastive learning. Our main contribution is to make use of the advantages of both inter-sample learning and intra-sample learning by introducing a novel framework for self-supervised video representation learning. Several options have also been explored towards the best practices in our framework. Further, for two commonly used evaluation tasks (i.e., video retrieval and video recognition), our method can surpass many existing state-of-the-arts by a notably large margin.

Our contributions are summarized as follows:

- We generate intra-negative samples by breaking temporal relations, which can extend negative samples and encourage the model to learn rich temporal information as well as spatial information for better video representation.

- We propose inter-intra contrastive learning, which is the most distinctive part that can make the most use of available data in self-supervised learning.
- We prove that many techniques such as data modality, data transformations, and head projector are generally effective in video self-supervised learning, which can be easily applied to other methods in this area.
- Extensive experiments show that with our IIC framework, significant improvements over contrastive learning baselines as well as other state-of-the-art methods can be achieved with the same network architecture.
- Our proposed inter-intra contrastive learning framework is flexible to be applied to other self-supervised contrastive learning methods.

3.2 Related Works

In this section, based on the learning style, we divide the existing self-supervised learning methods into two categories, inter-sample learning and intra-sample learning. The difference is whether the training depends on distinguishing differences from different video samples. Because we focus on video representation, another subsection is used to briefly introduce techniques in network backbones of video understanding. Besides, we also used one subsection to review other works in sampling strategy in self-supervised learning.

3.2.1 Intra-Sample Learning

For intra-sample learning methods, there is no interaction or constraints between different instances. And usually dedicatedly designed tasks are used to constrain input data, where different transformation functions are applied and models are optimized to recognize which have been done.

Self-supervised learning methods are close to unsupervised representation learning, including autoencoders [97] and variational autoencoders [98], trying to learn features by reconstructing data. Noroozi et al. [93] proposed to learn features by solving Jigsaw puzzles. In [94], context inpainting was used to train models. In [99], images were rotated and the models were trained by predicting the rotated angles. Features learned from these tasks can be transferred to image tasks with good performance. Guo et al. designed a self-correction module to co-train networks

in previous stages for hand pose estimation [100]. Xu et al. proposed a set of pretext tasks specifically designed for sketches [101].

Compared to images, videos contain an additional temporal axis and temporal information lies among different video frames. Therefore, it is important to efficiently extract temporal information by dedicatedly designed tasks. Temporal information is highly related to temporal orders, and many existing works utilize temporal orders to train their networks [5, 23–25]. Misra et al. [23] trained a network by distinguishing whether several input video frames were in the correct order. Odd-one-out network (O3N) [24] was proposed to identify unrelated or odd video clips. An order prediction network (OPN) [25] was trained by predicting the correct order of input frames. Xu et al. [5] used several video clips with 3D convolutional networks to predict the order. In addition to focusing on the temporal order, Wang et al. [102] proposed regressing motion and appearance statistics to learn video representations. Kim et al. [103] proposed training models by completing space-time cubic puzzles. Recognizing transformations is another solution. One of several transformations such as spatial rotation and temporal shuffling had been conducted on input data and the VCP [3] method was designed to recognize which action has been applied. There are also many works trying to train the network and make it sensitive to video playback speed [26–30, 104]. The key idea is to use different frame sampling rates and train the network to recognize the different speeds of video clips.

Most of the aforementioned methods can be named as pretext task based-methods, which try to train the network by recognizing transformations, especially temporal transformations. Once the network can detect which temporal transformation has been applied, the model is believed to have the ability for temporal representation.

3.2.2 Inter-Sample Learning

For inter-sample learning methods, features from the same sample should be close to each other in the feature space. On the contrary, features from different samples should be far from each other. In this way, the constraints are between different samples.

In [105], frames from the same video were treated as the anchor and positive samples while frames from other videos were negatives. And the network was optimized by the triplet loss [106]. Ranking loss with the siamese network was used in [107]. Several deep metric learning methods [108, 109] were also proposed to help constrain. After contrastive loss [31] was proposed, contrastive learning became the mainstream method in self-supervised learning. Contrastive Predictive

Coding (CPC) [32] used sequential data to learn the future from the past. Deep InfoMax [33] and Instance Discrimination [34] learned to maximize information probability from the same sample. In Contrastive Multiview Coding (CMC) [2], different views (e.g., different color space, depth image) from the same sample are used as anchor and positives. MoCo [35] used a momentum encoder with a momentum-updated encoder to conduct contrastive learning. And the video version of MoCo was presented in [110]. Different data augmentation methods were proved effective in SimCLR [38] for paired samples. Li et al. designed a self-supervised process and used pseudo labels to expand sample pairs in the loop self-supervised strategy [111]. Vu et al. designed Siamese architecture to train the contrastive feature extraction network for parking space status inference by analyzing images from parking lot [112]. There are also many recent works [37, 43, 91, 113, 114] trying to apply contrastive learning techniques to videos, which also belongs to the inter-sample learning category. For example, [114] is a video version of SimCLR with more explorations in videos. CoCLR [37] used optical flow data and formed co-training scheme. Since the main movements in videos usually exist in the foreground, Background Erasing (BE) [115] can be used to enhance the temporal information.

Contrastive learning usually requires positive-negative sample pairs to train. There are some very recent image-based unsupervised learning works that can obtain good performance without negative samples, such as BYOL [39], SimSiam [116]. And BYOL has been applied to video-based self-supervised learning in [110], together with some other contrastive learning methods.

3.2.3 Video Representation

Previous self-supervised learning methods have mainly been applied to images. Some video representation learning methods still use single or separate image frames as inputs [23–25], which do not benefit from new techniques related to video understanding.

For video representation, different network architectures have been proposed in supervised video recognition. In Temporal Segment Networks (TSN) [48], one video was divided into several segments, and frames from each segment are set as the input of a 2D CNN. In addition to the RGB data, two-stream ConvNets [21, 53, 54] have been used with an additional optical flow stream. Two-stream methods have also been boosted using dictionary learning [20] and semantic cues with a multi-scale strategy [117]. Recently, many 2D CNN based methods specially designed some modules to make use of temporal information, such as temporal relation

network [69], non-local blocks [51], temporal shift module (TSM) [50], temporal bilinear modeling [49] as well as temporal excitation and aggregation (TEA) block [59]. Spatio-temporal convolution neural networks (3D-CNNs) were also widely used in video recognition tasks. Tran et al. proposed C3D [6] which used several 3D convolutional layers and achieved good performance in video recognition. 3D convolutional versions of ResNet [60] and Inception net [62], R3D [8] and I3D [9], were proposed and showed promising performance on benchmark datasets [9, 56, 57]. In R(2+1)D [7] and S3D [10], one 3D convolutional kernel was separated into two steps, a spatial part and a temporal part, to save parameters and achieved better performance. SlowFast network [11] used two pathways to jointly extract video features. These trained models were proved to be effective feature extractors and can be applied to other video-related tasks.

All the aforementioned models can be used as the network backbone in self-supervised video representation learning. By replacing 2D CNN with 3D CNN, [5] reported better performance than [25] while their target tasks were the same, predicting the temporal orders of inputs. In [3, 5], C3D, R3D, and R(2+1)D were used to prove the effectiveness as well as the generality of their methods.

3.2.4 Sample Selection

For contrastive learning, positive and negative pairs are used to calculate the loss. The quality of sample pairs will affect the training. In most traditional contrastive learning methods, positive samples came from the same source instance (e.g., image or video) as the anchor while negative samples come from a different source. This can work well because the source instance is treated as a whole, meaning features representing scenes, objects as well as actions are similar in one source instance. Cross-modal contrastive learning methods [40, 118] usually used video frames and sounds together, where whether frames and sounds were correctly aligned was used as a supervision signal. When it came to frame pairs considering the temporal correspondence, better samples could be used in contrastive learning [119]. Positive samples could be selected by feature mining k-nearest neighbor search [113] or augmented from web data by filtering noises [120–122]. There are also a few works discussing issues around the selection of negatives in contrastive self-supervised learning, making use of feature distances [123], sample mixture [124] or other sampling strategies [125, 126] for better selection of negative samples.

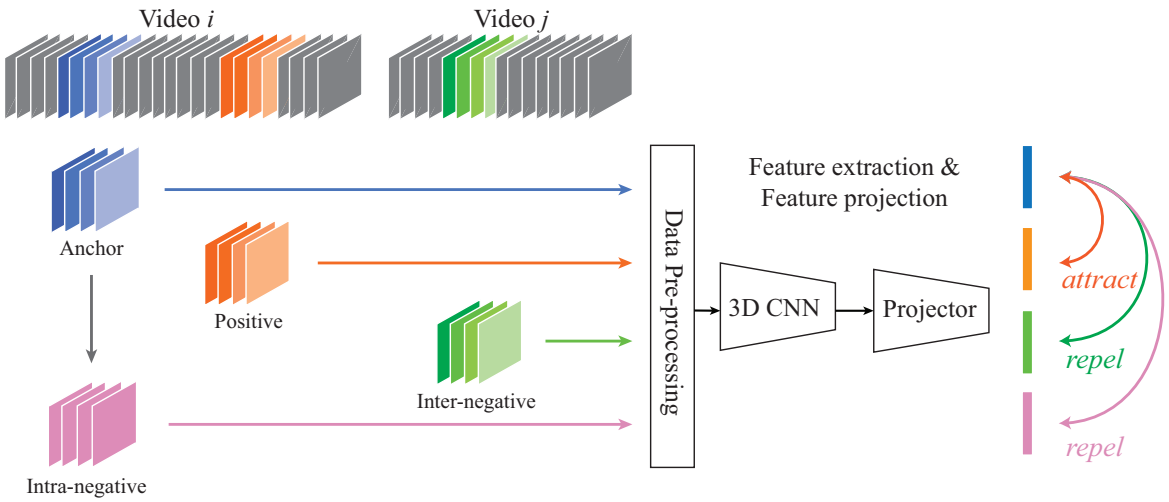


Figure 3.2: The main framework of IIC. Intra-negative samples are generated from the first view by breaking its temporal relationship. Video clips are transformed by data pre-processing strategies such as converting to residual clips, applying strong data augmentations. A two-layer MLP is applied to project features extracted from the network backbone to the target feature space. A contrastive loss is used for the optimization of the network.

3.3 Methods

Our goal is to learn discriminative feature representations from videos, not only for distinguishing one action from another, but also for capturing rich temporal information. The framework of our IIC is shown in Fig. 3.2. In this section, we start from the novel input part and then elaborate on contrastive learning with these inputs.

3.3.1 Inter and Intra Inputs

Considering different video clips from videos, if these video clips are from the same video, we can treat them as intra-samples. When video clips are from different videos, we call them inter-samples. Regarding whether they represent the same action, both intra-samples and inter-samples can be divided into positives and negatives. Inter-positives are not available in unsupervised learning because we do not know action labels for each video. And intra-positives are surely positive samples since they are from the same video instance, representing the same action. In contrastive learning, inter-samples are usually treated as negatives together with intra-positives.

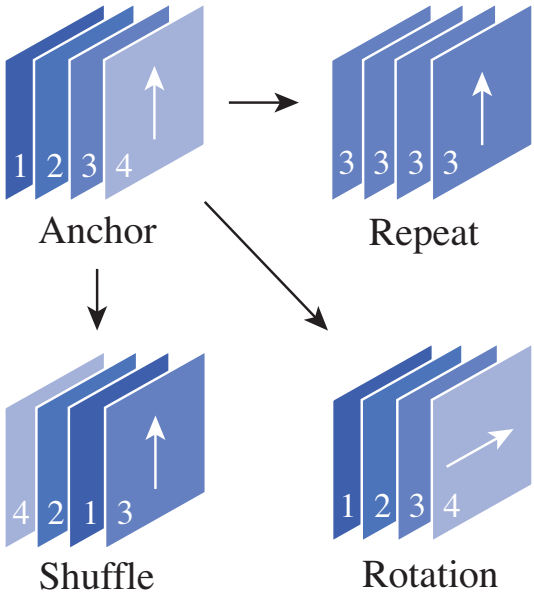


Figure 3.3: Generating intra-negative samples from original video clips.

Our proposed method extends negatives with intra-negative samples to make the most of the data for contrastive learning.

We denote a video set as X and two different sampled video clips (e.g., different data modalities, different sampling locations, different augmentation transformations) of videos as x_i^1 and x_i^2, where i is the video ID. A 3D convolutional network is used as the backbone to extract features and feature v_i^1 and feature v_i^2 can be obtained respectively. Therefore, the referred data x_i^1 and x_i^2 are in shape $THWC$, where T successive frames with height H and width W are stacked together. C is the channel number. Temporal information relies on the connections among T stacked frames. To clarify, in the following parts, we use "view" to refer to different sampled video clips and when we refer to view 1 and view 2, they represent positive pair samples (i.e., x_i^1 and x_i^2, v_i^1 and v_i^2).

Because the source is the same video i, feature v_i^1 and feature v_i^2 should be similar in the feature space. At the same time, feature v_i^1 should be different from features v_j^1 (for $j \neq i$) since they are from different videos. This kind of method is effective enough for images. However, video data have one more dimension. When the same person behaves in opposite ways, e.g., *standing up* and *sitting down*, the appearance information of each frame is similar, in which condition traditional contrastive learning methods can be easily fooled.

Here, we introduce intra-negative samples in contrastive learning for videos by breaking the temporal relationship. For one video clip, the data x_i^1 is a set of

frames. To simplify, we use $\{frame_1, ..., frame_T\}$ to represent a set of temporally-ordered frames. Three different intra-negative generation methods, frame repeating, temporal shuffling, and clip rotation, are proposed to break the temporal relationship and generate intra-negative video clips (Fig. 3.3). Though similar transformations might have been used in other works, we first use them to generate negative samples in video self-supervised learning.

Frame repeating. One frame that is randomly selected from the video clip is repeated T times to generate intra-negative samples (Eq. 3.1). Then no frame changes exist in this video clip and the corresponding temporal information should have been broken, even though the spatial information is almost the same as its source.

$$x_{repeat} = \{frame_k, ..., frame_k\},\ k = random(1, T). \tag{3.1}$$

Temporal shuffling. In the original video clip, frames are in the correct order. If these frames are randomly shuffled (Eq. 3.2), the actions will be strange and the corresponding action information should be different. Temporal shuffling does not change the global statistical information. And temporal clues play an important role for models to distinguish this kind of intra-negative samples from the source.

$$x_{shuffle} = shuffle(x),\ where\ x_{shuffle} \neq x. \tag{3.2}$$

Clip rotation. Rotation is one pretext task that is used in self-supervised learning [4, 99]. In videos, when one video clip is rotated using Eq. 3.3, where the angle θ is large, the movement direction is changed. In such cases, the rotated video clip should represent a different motion from the original one.

$$x_{rotation} = rotate(x, \theta),\ where\ \theta \neq 0. \tag{3.3}$$

Note that intra-negative samples can be generated for both anchor and intra-positive, and all the generating functions can be used simultaneously. In this work, only one generating function is used for each experiment because we found a combination of all intra-negative generation functions might not help (details can be found in Sec. 3.5.1). Then x^{neg} (either x_{repeat}, $x_{shuffle}$, or $x_{rotation}$) is used to represent an intra-negative sample from x^1. We also want to address that the generated intra-negative samples share similar pixel value distributions with the original one (Fig. 3.4). From the figure, we can find that the pixel value distributions for the anchor (Video1: view1), the positive (Video1: view2), and intra-negative samples (Video1:

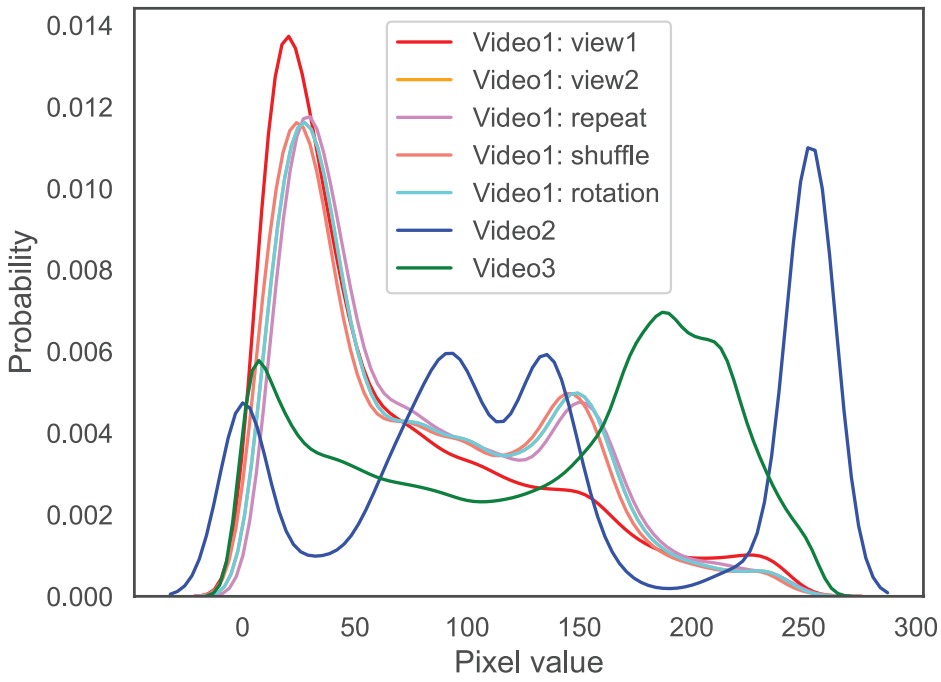


Figure 3.4: Distributions of statistical information of video clips. The first frame of the video clip is used because frame shuffling and rotation do not change global statistical information among one video clip. Intra-negative generation functions (i.e., frame repeating, frame shuffling, and frame rotation) maintain most or all global statistical information, which are applied to view 2 . Frames from the same video (Red and orange curves) share similar distributions while frames from videos (Red/orange, blue, and green curves) vary from each other.

repeat, Video1: shuffle, Video1: rotation) are close to each other, constraining the model to learn more discriminative temporal information from video clips.

3.3.2 Contrastive Learning

Contrastive learning uses anchor, positive, and negative samples and aims to extract discriminative features from the anchor and negative samples while maintaining the similarity between the anchor and positive samples. In traditional contrastive learning methods (e.g., CMC [2]), the sample pairs $\{x_i^1, x_i^2\}$ are positives while $\{x_i^1, x_j^2\}(i \neq j)$ are negatives. Because intra-negative samples are used in our approach, the negative pairs are extended by adding $\{x_i^1, x_j^{neg}\}$, where j can be equal to i.

A discriminative function $h_\theta(\cdot)$ is used to ensure that positive pairs have high values while the value for negative pairs should be low. The function is trained by selecting a single positive sample from a set of data. After feature v_i^1 has been

extracted, traditional contrastive learning methods train this function to correctly select a positive sample out of a set $S^2 = \{v_1^2, ..., v_i^2, ..., v_{k+1}^2\}$, which contains one positive sample v_i^2 and k negative samples. In our proposed method, another set $S^{neg} = \{v_1^{neg}, ..., v_{k+1}^{neg}\}$ is also used that only contains negative samples. The loss function is similar to recent works for contrastive learning [2, 32, 127]:

$$\mathcal{L}_{contrast}^{v_i^1} = -\log \frac{h_\theta(\{v_i^1, v_i^2\})}{\sum_{j=1}^{k+1} h_\theta(\{v_i^1, v_j^2\}) + \sum_{j=1}^{k+1} h_\theta(\{v_i^1, v_j^{neg}\})}. \tag{3.4}$$

Here, k is the number of negative samples, which can be equal to $N-1$, where N is the total number of training samples. We randomly select k samples from N where $k \ll N$ to accelerate training. Memory bank [34] is used to save and process these features, and the sample sets S^2, S^{neg} can be treated as subsets of features of corresponding memory banks M^2, M^{neg}. In recent works, contrastive learning methods can use a queue [35, 108] to save previous computed features or just use larger batch sizes [38] to get more negative samples. In our case, with the projection head (Sec. 3.3.3), features are in 128 dimensions for every sample in the memory bank, and the total memory consumption for the Kinetics-400 dataset is less than 120 MB, whose memory consumption is much smaller than methods that use momentum encoder [35, 39].

The updating procedure of the memory bank is as follows,

$$M_i = \mu v_i + (1-\mu) M_i, \tag{3.5}$$

where M_i is the i_{th} feature in the memory bank to record which sample features belong to, and μ is the momentum decay weight.

The critic $h_\theta(\cdot)$ is implemented by feature representations using the non-parametric softmax technique [34]. Then we can compute this function as in the following:

$$h_\theta(\{v_i^1, v_j^2\}) = \exp\left(\frac{v_i^1 \cdot v_j^2}{\|v_i^1\| \cdot \|v_j^2\|} \cdot \frac{1}{\tau}\right), \tag{3.6}$$

where τ is a hyper-parameter that controls the range of the results. In our work, three memory banks are used to store video features from previous iteration, and these features can be fetched as non-parametric weights when calculating the loss [34].

Eq. 3.4 only treats view 1 as an anchor. When treating view 2 as an anchor, symmetrically, another loss can be calculated and they are added to form the final

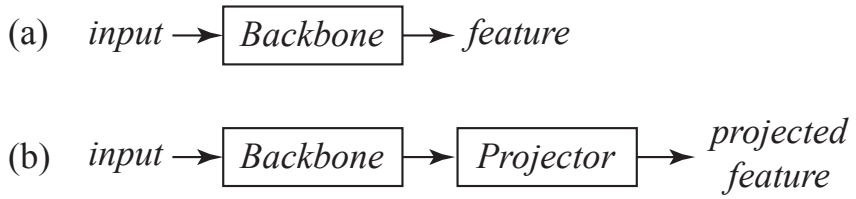


Figure 3.5: (a) Contrastive learning uses features directly from the backbone. (b) An additional projector network (two-layer MLP) is used to project features to another feature space for contrastive learning.

loss function:

$$\mathcal{L} = \mathcal{L}^{v^1}_{contrast} + \mathcal{L}^{v^2}_{contrast}. \tag{3.7}$$

3.3.3 Data Strategies

Data modalities. In video recognition, the most widely used data modalities are traditional RGB input and the optical flow. These two data modalities have also been set as two common views in contrastive learning [128, 2]. However, optical flow data usually require additional calculation and storage.

In video tasks, frame difference is another data modality and has also been used in existing works [48] with 2D ConvNets. Residual frames with 3D ConvNets have been proved to be more effective compared to original RGB video clips in Sec. 2. We adopt residual clips because its supreme performance in this area. The calculation of residual clips is

$$ResClip = \{frame_{i+1} - frame_i, \cdots, frame_{i+T+1} - frame_{i+T}\} \tag{3.8}$$

Augmentation transformations. In previous methods in video self-supervised learning, only a few data augmentations were conducted. Some recent works [35, 39] started to use strong augmentations in images, which have achieved improvements over the corresponding baselines.

Though these data augmentations are conducted on images, this kind of processing can be easily applied to video frames. Another motivation is that we wonder whether it will also boost the performance of residual clips because there will be much less color information in residual clips.

Projection head. In some previous contrastive learning works such as CMC [2], the features are directly from the feature encoding network without a projection

Algorithm 1: Training with IIC framework

Require: Video data: *Dataloader* which contain v_i^1, v_i^2, i;
Network: backbone *net*, projection network net_p;
Memory banks: $M = \{M_1, M_2, M_{neg}\}$ % Can be replaced using a queue to save features as MoCo [35]
Ensure: Optimized network parameters: *net*
1: Initialize network net, net_p and memory banks Ms
2: **for** $iter = 1 \rightarrow max\_iteration$ **do**
(a) Fetch data $x_i^1, x_i^2, i = load(Dataloader)$.
(b) Generate intra-negative samples $x_i^{neg} = f(x_i^1)$.
(c) Apply data pre-processing strategies to x_i^1, x_i^2, x_i^{neg}.
(d) Extract video features $v = net(x)$.
(e) Project features to contrastive space $v_p = net_p(v)$.
(f) Calculate *loss* using Eq. 3.4 and optimize *net*.
(g) Update memory bank M with corresponding feature v_p using Eq. 3.5.
endfor
3: **return** *net*;

head. Recent works [36, 38] utilized a projection head to project features to another feature space and conducted contrastive learning based on projected features. In our case, the target is to make use of contrastive learning and train a network to extract good temporal features. However, effective features vary from one task to another. For example, the downstream task video retrieval used these features directly from the backbone while for action recognition, results come from the classifier. The optimization of self-supervised learning can be treated as another task, where feature isolation using a projection head can help the network backbone focus on more general video features, instead of particular features for the contrastive learning only. Similar approaches can be found in multi-task learning methods [129, 130], where different supervision signals are used from different sub-branches. Therefore, we also used an additional projection head in this work to form the final solution (Fig. 3.5 (b)).

To summarize this section, we write the process flow of our proposal in Algorithm 1.

3.4 Experiments

Extensive experiments were conducted to evaluate our proposed IIC framework in different tasks and datasets. Because there are several options as well as technique strategies in our framework, we first elaborate on the effectiveness of some settings and then apply these settings to compare with other methods.

3.4.1 Datasets

For self-supervised video representation learning, datasets are usually come from those for supervised video recognition, such as UCF101 [56], HMDB51 [57], and Kinetics400 [9]. However, when using these datasets, labels are discarded to form the unsupervised learning scheme. The UCF101 dataset contains 13,320 videos, which consists of 101 different action categories. HMDB51 consists of 6,849 videos containing 51 action classes. Kinetics400 is a much larger dataset, consisting of around 240k videos.

For fair comparisons with existing works [2, 3, 5], we followed their settings and mainly used UCF101 dataset to conduct self-supervised training part and used UCF101 and HMDB51 datasets for evaluation. Both UCF101 and HMDB51 datasets have three data splits. And if not specially declared, results are averaged over three splits. Self-supervised learning part can be treated as the pre-training period. And larger dataset can bring further improvements. Thus, we also used the Kinetics400 dataset to pre-train our network for further improvements.

3.4.2 Evaluation Tasks

Because it is a self-supervised learning method, models can be directly used once training is completed without fine-tuning on other tasks. Therefore, the trained models are used to extract video features and then the performance on video retrieval is evaluated. UCF101 and HMDB51 are two different datasets. We trained our model only on UCF101 *split* 1 and performed video retrieval on both UCF101 and HMDB51 datasets. To evaluate whether video recognition can benefit from our self-supervised learning method, we also conducted experiments by fine-tuning trained models on both UCF101 and HMDB51 datasets.

3.4.3 Options in the Framework

Multiple modalities. For video representation learning, traditional RGB input and the corresponding optical flow were set as two common views [2, 128]. Because calculating optical flow requires additional computation, in this work we try to find the most effective views for contrastive learning in videos without optical flow.
Backbone networks. 3D CNNs have been proved to be more powerful than 2D CNNs to extract motion features from videos [7–10]. Recent self-supervised video representation methods [5, 3] used C3D [6], R3D [8], and R(2+1)D [7] as their network backbones. For fair comparison, we follow their settings to conduct experiments. In addition, results using R3D-18 [7] and S3D-G [10] are also reported.
Intra-negative generation. As we discussed in section 3.3.1, we introduce three intra-negative generating methods, namely frame repeating, temporal shuffling, and clip rotation. In our experiments, only one intra-negative generation method is used in each experiment. We have tried different combinations of them together, but the performance is not as good as using them alone, which will be introduced in ablation studies.

3.4.4 Implementation Details

We mainly follow [6] for data preparation. Sixteen successive frames of size 128×171 are stacked together to form a video clip. Random spatial cropping is conducted to generate input data of size $16 \times 112 \times 112$, where the channel number 3 is ignored. For residual clips, the original RGB video clip is shifted along the temporal axis and the difference between the original clip and the shifted clip is the corresponding residual clip. Only one 3D ConvNet is used to cope with different views of input data. For the S3D-G network, we also used input data in size of $64 \times 224 \times 224$ to compare.

The feature projector is composed of two fully-connected layers. Features extracted from the network backbone will be projected to 128 dimensions to calculate the contrastive loss.

For frame repeating, the repeated frame is randomly chosen. For temporal shuffling, the transformation is similar to [3]. We divide one video clip into four sub-clips, and shuffle the sub-clips to conduct shuffling. As to clip rotation, video clips are rotated 90 degrees.

The batch size is set to 16 and training lasts for 240 epochs for the self-supervised learning procedure. The initial learning rate is set to 0.01 and it is updated by the

cosine decay strategy [131]. In the non-parametric learning part, $2k$ negative samples are sampled from memory banks, with k set to 1,024. Video retrieval is conducted on the basis of video-level features, which are generated by averaging clip features from the same video. K-nearest neighbor search is used to calculate retrieval accuracy. When evaluating in video recognition, we use our trained models as an initialization strategy, and the learning rate is set to 0.001 for fine-tuning. The best performance on the validation dataset is used for testing.

3.5 Results and Analysis

In this section, we first report our ablation studies. Then, we compete with the state-of-the-art methods in self-supervised spatio-temporal learning. In this work, we evaluate the performance in two aspects: retrieval accuracy and recognition accuracy. Video retrieval is conducted using video-level features, which are averaged by clip-level features from the same video. Video recognition is conducted by fine-tuning the self-supervised pre-trained models.

In addition to video frames, some existing works use additional modalities like audio [41, 130, 132–134], or narrations [132], and train on much larger datasets with larger input size. We do not include them in tables for fair comparison.

3.5.1 Ablation Studies

We conduct ablation studies in many aspects. If not specified, all the ablation studies are based on the R3D backbone and the intra-negative generation method is clip rotation. Results are reported on UCF101 split 1 in video retrieval and recognition. Head projection is used to project features to contrastive space as well as reducing feature dimensions.

Modality Choices

We try to find the most efficient way to use RGB frames, without additional computation or complexity for pre-computed features such as optical flow. Therefore, we tried different combinations between RGB clips and residual clips. Although traditional contrastive learning in videos used two RGB video clips as positive/negative pairs and in [91], experiments showed that setting RGB clips and residual clips as paired samples can obtain better performance than RGB clips. Here we also tried to

Table 3.1: Ablation studies on video modalities. R3D is used as the network backbone and rotation is used to generate intra-negative samples. Results are reported on UCF101 split 1.

View1	View2	Top1	Top5	Recognition
RGB	RGB	40.4	54.0	62.0
RGB	Res	52.8	72.0	75.4
Res	Res	53.1	70.1	77.8

Table 3.2: Ablation studies on head projector. R3D is used as the network backbone. Results are reported on UCF101 split 1.

Modality	Intra-neg	Head	Top1	Top5	Recognition
Res	×	×	43.5	61.1	74.2
Res	shuffle	×	40.7	56.3	74.0
Res	repeat	×	45.2	62.1	74.3
Res	rotate	×	**46.4**	**63.9**	**74.9**
Res	×	✓	50.4	68.5	76.4
Res	shuffle	✓	49.4	65.4	76.5
Res	repeat	✓	53.0	68.2	77.2
Res	rotate	✓	**53.1**	**70.1**	**77.8**

sample two residual clips from the same video as different views. Results are shown in Table 3.1.

As we can see from the table, when two views are both RGB clips, the top-1 retrieval accuracy is 40.4% and the recognition accuracy is 62.0%. When using RGB and residual clips as two different views, the performance for video retrieval is much better, which corresponds to the conclusion from CMC [2] as well as our previous work [91]. When using residual clips for both views, the best performance can be achieved, reaching 53.1% at the top-1 retrieval accuracy and 77.8% for recognition.

Thus, for the rest of our experiments, we mainly use residual clips for input data.

Projection Head

As we introduced, in many works [2, 35], the projector is not used while in some recent works [36, 38, 135], better performance can be achieved with a projection

Table 3.3: Ablation studies on data augmentation transformations. R3D is used as the network backbone. Results are reported on UCF101 split 1.

Modality	Intra-neg	Aug.	Top1	Top5	Recognition
RGB	×	×	14.8	25.5	50.6
Res	×	×	27.0	44.6	72.5
Res	repeat	×	31.5	49.6	72.8
RGB	×	✓	40.3	55.7	61.7
Res	×	✓	50.4	68.5	76.4
Res	repeat	✓	53.0	68.2	77.2

head. These works are all image-based self-supervised methods. We here conducted ablation studies on the effects of the projection head in video-based representation learning.

In Table 3.2, we show four comparison pairs. With the projection head, better retrieval performance can be achieved in all three settings. For video recognition, which does not use any parameters in the projection head, at least 1.5% points improvements can be achieved.

Based on these findings, in the following experiments, we adopt the projection head as one default setting.

Data Augmentation

Data augmentation is proved to be effective in images [36, 38]. However, in self-supervised video representation learning, previous works [3, 5, 28] did not use this kind of strategy. Here we show some experimental results in Table 3.3.

From the table, we can see that both RGB modality and residual modality can enjoy the benefit of strong data augmentations. With strong data augmentations, the worst retrieval performance in these three settings is 40.3%, even better than the best of these without strong data augmentation. The best recognition performance is obtained by using the intra-negative strategy with residual modality, reaching 77.2%.

Therefore, in our following experiments, we use strong data augmentations as one default setting.

Table 3.4: Ablation studies on Intra-negative types. R3D is used as the network backbone. Results are reported on UCF101 split 1.

Intra-neg	Top1	Top5	Recognition
Baseline	50.4	68.5	76.4
Repeat	53.0	68.2	77.2
Shuffle	49.4	65.4	76.5
Rotate	**53.1**	**70.1**	**77.8**
Repeat + Shuffle	44.6	62.7	76.8
Repeat + Rotate	43.1	61.6	76.5
Shuffle + Rotate	50.8	68.0	76.0
All neg	42.5	60.5	74.4

Intra-Negative Generation

We introduced three types of intra-negative sample generation functions: frame repeating, frame shuffling, and clip rotation. We show the performance of each in both video retrieval and recognition in Table 3.4. Without using any intra-negative samples, the top-1 accuracy for video retrieval is 50.4% and for recognition, the performance is 76.4%. With frame repeating or clip rotation, the top-1 accuracy for video retrieval can be increased by over 2.6% points and the best performance for video recognition is achieved by using frame rotation. Frame shuffling is not as effective as the other two intra-negative generation functions, the retrieval accuracy is even lower than the baseline. It may depend on the experimental settings as frame shuffling has shown effectiveness in [91]. Based on the results, among all these three intra-negative generation functions, clip rotation is the best choice and frame shuffling is the last one. However, the gaps are small. Therefore, we adopt all these three settings as optional configurations in the main comparisons with state-of-the-art methods.

We also want to mention that when combining different intra-negative generation methods, some performances are even poorer than the baseline method for video retrieval and recognition. We also consider that introducing different intra-negative generation methods will greatly increase the difficulty during training because there are too many hard-negative samples at the same time. Training for more epochs might help. However, under the current empirical experimental settings, using

Table 3.5: Comparison with state-of-the-art methods in video retrieval on UCF101 split 1. † indicates methods using optical flow in the training period. We highlight the best results in each block in **bold**.

Methods	Backbone	Top1	Top5	Top10	Top20	Top50
MemDPC [37]	R2D3D	20.2	40.4	52.4	64.7	-
MemDPC-Flow† [37]	R2D3D	40.2	63.2	71.9	78.6	-
CoCLR-RGB† [113]	S3D	**53.3**	**69.4**	**76.6**	**82.0**	-
VCOP [5]	R3D	14.1	30.3	40.0	51.1	66.5
VCP [3]	R3D	18.6	33.6	42.5	53.5	68.1
PRP [28]	R3D	22.8	38.5	46.7	55.2	69.1
IIC (repeat)	R3D	53.0	68.2	75.1	81.5	88.3
IIC (shuffle)	R3D	49.4	65.4	72.2	79.7	87.3
IIC (rotate)	R3D	**53.1**	**70.1**	**77.4**	**84.0**	**91.4**
VCOP [5]	R(2+1)D	10.7	25.9	35.4	47.3	63.9
VCP [3]	R(2+1)D	19.9	33.7	42.0	50.5	64.4
PRP [28]	R(2+1)D	20.3	34.0	41.9	51.7	64.2
PacePred [30]	R(2+1)D	25.6	42.7	51.3	61.3	74.0
IIC (repeat)	R(2+1)D	**51.6**	67.8	74.8	81.0	88.5
IIC (shuffle)	R(2+1)D	50.3	65.5	73.2	79.8	87.9
IIC (rotate)	R(2+1)D	50.6	**68.3**	**76.0**	**82.9**	**90.5**

any one of these three methods can usually perform better. For this part, further discussions and possible explanations are in Sec. 3.6.4.

3.5.2 Comparison: Video Retrieval

For a fair comparison, we followed the settings in previous works [3, 5, 28, 91] and trained our models on UCF101, and tested them on both UCF101 and HMDB51 datasets. Video-level retrieval performance is reported here. Clip-level features are averaged to represent the corresponding video, and a k-nearest neighbor search is conducted. When the retrieved video has the same action label as the target video, one hit is confirmed.

The results on the UCF101 dataset are shown in Table 3.5 and Table 3.6. Most compared methods require a dedicatedly designed task to train the model, which belongs to the intra-sample learning category. MemDPC [37] and PacePred [30] utilized contrastive learning technologies, treating every different samples as negative. As shown in this table, the top-1 accuracy for IIC was already higher than most other works. Except for our work, the best performance for video retrieval is 53.3%,

Table 3.6: Comparison with state-of-the-art methods in video retrieval on UCF101 split 1 using C3D and R3D-18. We highlight the best results in each block in **bold**.

Methods	Backbone	Top1	Top5	Top10	Top20	Top50
VCOP [5]	C3D	12.5	29.0	39.0	50.6	66.9
VCP [3]	C3D	17.3	31.5	42.0	52.6	67.7
PRP [28]	C3D	23.2	38.1	46.0	55.7	68.4
PacePred [30]	C3D	31.9	49.7	59.2	68.9	80.2
IIC (repeat)	C3D	54.3	69.3	75.6	81.8	89.5
IIC (shuffle)	C3D	52.0	67.2	73.4	79.9	88.0
IIC (rotate)	C3D	**55.1**	**72.1**	**78.4**	**84.0**	**91.6**
3DRotNet [4]	R3D-18	14.2	25.2	33.5	43.7	59.5
VCP [3]	R3D-18	22.1	33.8	42.0	51.3	64.7
RTT [29]	R3D-18	26.1	48.5	59.1	69.6	82.8
PacePred [30]	R3D-18	23.8	38.1	46.4	56.6	69.8
IIC (repeat)	R3D-18	54.7	70.1	76.5	82.8	89.9
IIC (shuffle)	R3D-18	50.8	65.1	71.6	78.8	86.5
IIC (rotate)	R3D-18	**56.2**	**71.3**	**77.5**	**84.6**	**91.6**

Table 3.7: Comparison with state-of-the-art methods in video retrieval on HMDB split 1.

Methods	Backbone	Top1	Top5	Top10	Top20	Top50
VCOP [5]	R3D	7.6	22.9	34.4	48.8	68.9
VCP [3]	R3D	7.6	24.4	36.3	53.6	76.4
PRP [28]	R3D	8.2	25.8	38.5	63.3	75.9
IIC (repeat)	R3D	19.8	**43.1**	55.6	68.6	85.5
IIC (shuffle)	R3D	18.1	36.5	50.5	64.5	81.5
IIC (rotate)	R3D	**20.4**	**43.1**	**56.3**	**70.5**	**86.3**

obtained by CoCLR-RGB [113] using optical flow data in the training period. Without optical flow, the best result is 31.9% using PacePred [30] with C3D network backbone in Table 3.6. By using our IIC, the top-1 retrieval performance can be 56.2%, which does not use optical flow data but 26.0% points higher than MemDPC [37], and 24.3% higher than PacePred [30]. Also, we can find that all three intra-negative generation functions can achieve comparable results with each other, much higher than those from other works. Rotation seems to be slightly better than the other two intra-negative generation functions.

We test the transferability of the trained model on the HMDB51 dataset because even though no labels are used in the training dataset (i.e., UCF101), the testing dataset (i.e., HMDB51) is different from the training part. The results are shown in

Table 3.8: Comparisons with the state-of-the-art self-supervised methods on UCF101 and HMDB51 dataset.

Method	Date	Pre-train	ClipSize	Network	UCF	HMDB
Random		-	16×112^2	R3D	54.5	23.4
VCOP [5]	2019	UCF	16×112^2	R3D	64.9	29.5
VCP [3]	2020	UCF	16×112^2	R3D	66.0	31.5
PRP [28]	2020	UCF	16×112^2	R3D	66.5	29.7
IIC [91]	2020	UCF	16×112^2	R3D	74.4	38.3
IICv2 (repeat)		UCF	16×112^2	R3D	**78.6**	40.4
IICv2 (shuffle)		UCF	16×112^2	R3D	77.9	39.3
IICv2 (rotate)		UCF	16×112^2	R3D	**78.6**	**43.4**
Random		-	16×112^2	C3D	61.8	24.7
VCOP [5]	2019	UCF	16×112^2	C3D	65.6	28.4
VCP [3]	2020	UCF	16×112^2	C3D	68.5	32.5
PRP [28]	2020	UCF	16×112^2	C3D	69.1	34.5
IIC [91]	2020	UCF	16×112^2	C3D	70.0	30.8
RTT [29]	2020	K400	16×112^2	C3D	69.9	39.6
MoCo [115]	2020	UCF	16×112^2	C3D	60.5	27.2
MoCo + BE [115]	2021	UCF	16×112^2	C3D	72.4	42.3
RSPNet [104]	2021	K400	16×112^2	C3D	76.7	**44.6**
IICv2 (repeat)		UCF	16×112^2	C3D	78.1	39.6
IICv2 (shuffle)		UCF	16×112^2	C3D	78.2	39.4
IICv2 (rotate)		UCF	16×112^2	C3D	**78.5**	40.3

Table 3.7. A similar conclusion can be drawn. No matter what backbone to use, our IIC can achieve better performance than the other methods. With the R3D network backbone, the best performance was 20.4% at top-1 accuracy, surpassing the current state-of-the-art results by a large margin. By using other network backbones, the performances are similar, which we show in Appendix B.

3.5.3 Comparison: Video Recognition

Video feature representation is usually evaluated in the video recognition task. Here we initialized models parameters with weights from self-supervised learned model and the models were fine-tuned on two benchmark datasets, UCF101 [56] and HMDB51 [57]. We follow the majority settings in the pre-training dataset and video clip size, and tried four different network backbones. Because there are a lot of optional network backbones, we use two tables to show the results, illustrated in Table 3.8 and Table 3.9. All results are averaged over three splits.

From the table, we can see that better performances than a random initialization strategy can be achieved, revealing that better temporal information has been

Table 3.9: Comparisons with the state-of-the-art self-supervised methods on UCF101 and HMDB51 dataset. Results are averaged over three splits. † indicates methods using optical flow.

Method	Date	Pre-train	ClipSize	Network	UCF	HMDB
OPN [25]	2017	UCF	227^2	VGG	59.6	23.8
DPC [43]	2019	K400	16×224^2	R3D-34	75.7	35.7
CBT [42]	2019	K600+	16×112^2	S3D	79.5	44.6
SpeedNet [26]	2020	K400	64×224^2	S3D-G	81.1	48.8
MemDPC† [37]	2020	K400	40×224^2	R-2D3D	86.1	54.5
PacePred [30]	2020	K400	64×224^2	S3D-G	87.1	52.6
CoCLR (RGB)† [113]	2020	K400	32×128^2	S3D	87.9	54.6
STS [136]	2021	K400	64×224^2	S3D-G	**89.0**	**62.0**
IICv2 (rotate)		K400	16×112^2	S3D-G	83.6	48.0
IICv2 (rotate)		K400	64×224^2	S3D-G	88.6	55.2
Random		-	16×112^2	R3D-18	42.4	17.1
3D-RotNet [4]	2018	K400	16×112^2	R3D-18	62.9	33.7
ST-Puzzle [103]	2019	K400	16×112^2	R3D-18	65.8	33.7
DPC [43]	2019	K400	16×128^2	R3D-18	68.2	34.5
RTT [29]	2020	K400	16×112^2	R3D-18	79.3	**49.8**
RSPNet [104]	2021	K400	16×112^2	R3D-18	74.3	41.8
IICv2 (repeat)		UCF	16×112^2	R3D-18	**80.1**	41.2
IICv2 (shuffle)		UCF	16×112^2	R3D-18	75.2	38.2
IICv2 (rotate)		UCF	16×112^2	R3D-18	80.0	42.9
Random		-	16×112^2	R(2+1)D	55.8	22.0
VCOP [5]	2019	UCF	16×112^2	R(2+1)D	72.4	30.9
VCP [3]	2020	UCF	16×112^2	R(2+1)D	66.3	32.2
PRP [28]	2020	UCF	16×112^2	R(2+1)D	72.1	35.0
RTT [29]	2020	UCF	16×112^2	R(2+1)D	**81.6**	**46.4**
PacePred [30]	2020	K400	16×112^2	R(2+1)D	77.1	36.6
STS [136]	2021	UCF	16×112^2	R(2+1)D	77.8	40.7
IICv2 (repeat)		UCF	16×112^2	R(2+1)D	78.5	41.1
IICv2 (shuffle)		UCF	16×112^2	R(2+1)D	78.4	40.9
IICv2 (rotate)		UCF	16×112^2	R(2+1)D	78.5	42.5

embedded by self-supervised learning to some extent. When using R3D-18, R3D, and C3D as the network backbone, our IIC achieves the best performance among all these methods in the UCF101 dataset, including these methods which used the Kinetics-400 dataset to conduct self-supervised training. The total video length of the Kinetics-400 dataset is about 28 days while it is around 1 day for the UCF101 dataset. Usually, pre-training on larger datasets can help methods improve performance. With IIC, we also want to show that our methods are efficient and can even beat some methods which were pre-trained on Kinetics-400 while the size of the pre-trained dataset for ours is 3.6% of theirs. With R(2+1)D network, our methods also achieved better performance than others except for RTT [29]. However, the performances of

RTT are not stable such that its results using R3D-18 and C3D are 77.3% and 69.9% respectively, lower than our proposed methods. There are some very recent methods such as Background Erasing (BE) [115] and RSPNet [104]. With the same input size, our IIC can outperform RSPNet and BE when using C3D [6] as the network backbone.

When using Kinetics-400 dataset to pre-train, we show in Table 3.8 that our IIC can achieve 83.6% in UCF101 dataset with input size $16 \times 112 \times 112$ using S3D-G [10] network. With larger input sizes (e.g., $64 \times 224 \times 224$), the performance can be further boosted to 88.6%, revealing that besides network backbones, the resolution is also an important element when evaluating models. Under fair comparison, our IIC can outperform SpeedNet [26] and PacePred [30] in the same settings. Though the method STS [136] can obtain better performance in this setting, the performance using R(2+1)D is worse than ours regardless of which intra-negative generation function to use.

The transferability was again tested on the HMDB51 dataset, which is more complicated because this is not only transferable for different tasks, but also on different datasets. With the improved version of IIC, we can obtain 4.2% points improvements (74.4% to 78.6%) over our previous version using R3D on the UCF101 dataset. And with our IIC, we can obtain state-of-the-art performance based on R3D and S3D-G network architecture. When using other network backbones such as R3D-18, C3D, and R(2+1)D, our proposed methods can also achieve better results than others in most cases. Though the performance of RSPNet [104] and RTT [29] can achieve better performance in particular conditions (e.g., RSPNet achieved better performance with C3D in HMDB51, RTT achieved better performance with R(2+1)D), their results are not stable when using a different network backbone (e.g., RSPNet with R3D-18, RTT with C3D). Therefore, we could say that our IIC performs generally better than other methods in most cases.

Because we have three kinds of intra-negative generation functions in total, the performance of each intra-negative generation function is also listed in the table. From the recognition performance, it might be hard to say which is better among these three methods because the gaps are very small in the UCF101 dataset. For the results in the HMDB51 dataset, we might say clip rotation is the best intra-negative generation function.

Therefore, generally speaking, clip rotation is the best choice among these three intra-negative generation functions while frame shuffling is the worst one. Some performances in video retrieval and recognition might not be consistent with this

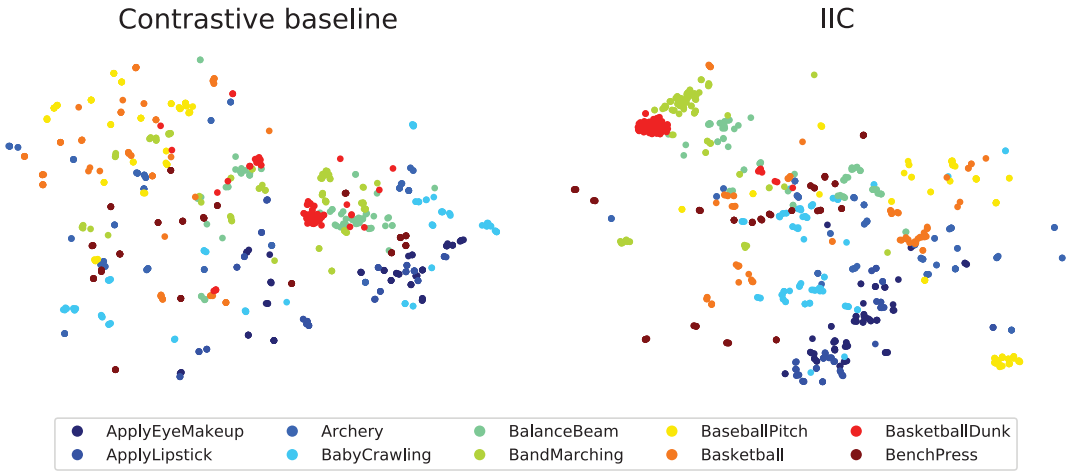


Figure 3.6: Feature visualization by t-SNE. Features extracted by IIC are more semantically separable compared to directly applying contrastive learning videos [2]. Each video is visualized as a point, with videos belonging to the same action category having the same color.

conclusion when using different backbones, and the reason might lie in the situation that we do not explore the best training hyper-parameters for each case.

3.6 Discussions

In this section, we will pose further discussions and provide more pieces of evidences on the advantage and the potential reasons why our proposal can help extract better temporal clues.

3.6.1 Visualization: Feature Embedding

Before comparing our proposed method with other state-of-the-art methods, we set the trained models as feature extractors and qualitatively evaluated video features by visualization in order to verify whether good feature representations have been learned. Here the first 10 categories (arranged by action names in alphabetical order) in the UCF101 dataset were used. Features were projected to two-dimensional space using t-SNE [137]. Fig. 3.6 visualizes the embedding of the features extracted by traditional contrastive learning [2] and our IIC. The contrastive baseline is a simple application of image-based contrastive learning methods to videos without considering the particularity of temporal information. It is obvious that IIC shows

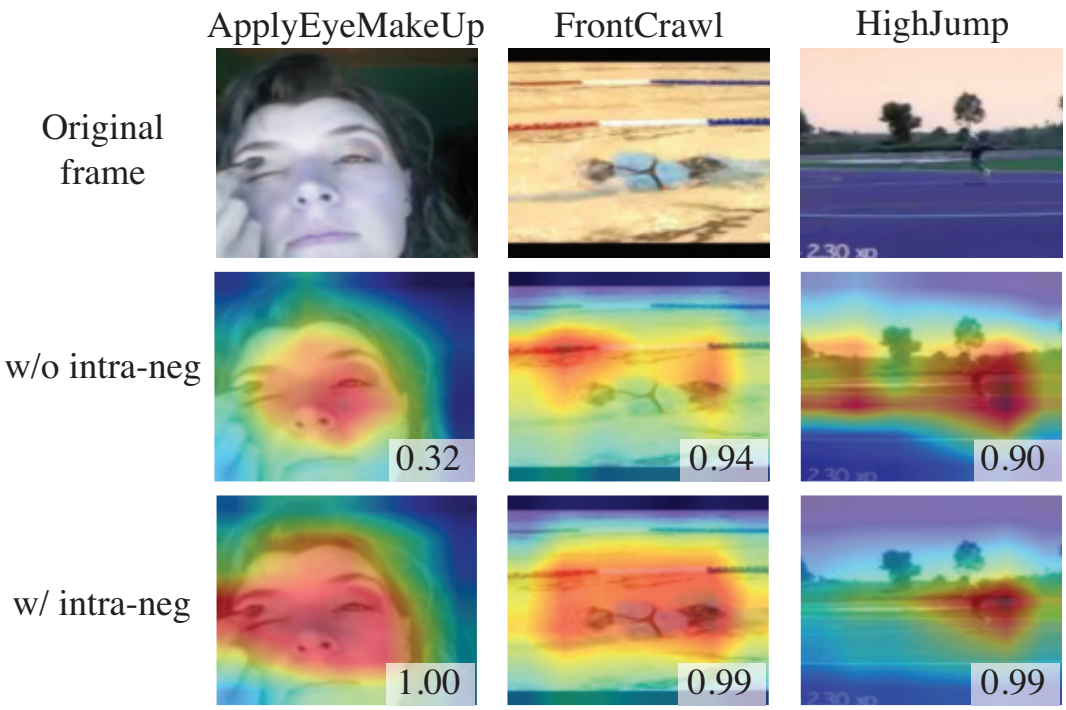


Figure 3.7: Class activation map visualization using Grad-CAM [1]. With the proposed intra-negative samples, the network will focus more on the moving part/entity instead of the background.

better feature clustering ability for video data, which reveals that better video representations can be learned by our IIC.

3.6.2 Visualization: Activation Map

To better understand the learned clues of IIC, we use the class activation map technique Grad-CAM [1] to visualize the region of interest. As we can see from Fig. 3.7, without intra-negative samples, though the network can still make the right judgment, the effective clues are mainly from the still background such as the swimming pool for category *FrontCrawl* or the track field for category *HighJump*. With our proposed intra-negative samples, the network will focus more on the moving part/entity, such as the moving hand for category *ApplyEyeMakeUp* or the athlete for the other two samples.

Table 3.10: Different ways to treat generated samples. Performances are reported in video retrieval and action recognition tasks. The "baseline" has already used our proposed strategies.

Method	Treated as	Top1	Top5	Recognition
Baseline	-	50.4	68.5	76.4
Repeat	Positive	45.3	61.8	76.3
Repeat	Negative	53.0	68.2	**78.6**
Shuffle	Positive	45.0	60.2	75.3
Shuffle	Negative	49.4	65.4	77.9
Rotate	Positive	38.9	55.1	75.8
Rotate	Negative	**53.1**	**70.1**	**78.6**

3.6.3 Potential Mechanism of Intra-Negative Samples

For the used transformation functions in our IIC, frame repeating and frame shuffling will generate video clips with abnormal sequence orders, and clip rotation will change the action directions. All these transformations will break temporal information more or less. Though some information can still remain, from intuition, it should help extract temporal features when treating these samples as negatives than as positives. We conducted experiments to compare these situations in Table 3.10. It is clear to get the conclusion that when treating generated samples as positives, it violates the rule of contrastive learning because the anchor and the generated video clip do not share similar temporal information, decreasing the performance of the baseline. When treating them as negatives, it benefits the model and helps the model capture more temporal information, resulting in better performances in two downstream tasks, though frame shuffling is the only exception when it comes to video retrieval task.

The other proof is obtained when calculating feature distances before self-supervised learning period. The features distances are calculated from the anchor and another feature from intra-positive, intra-negative, or inter-negative sample. As we can see from Fig. 3.8, even though the network parameters are randomly initialized, the overlap between traditional positive pairs (anchor and intra-positive, the blue curve) and negative pairs (anchor and inter-negative, the orange curve) is small, indicating that it would be very easy to distinguish positive and negative samples. The distances between the anchor and shuffled clips (purple curve) are

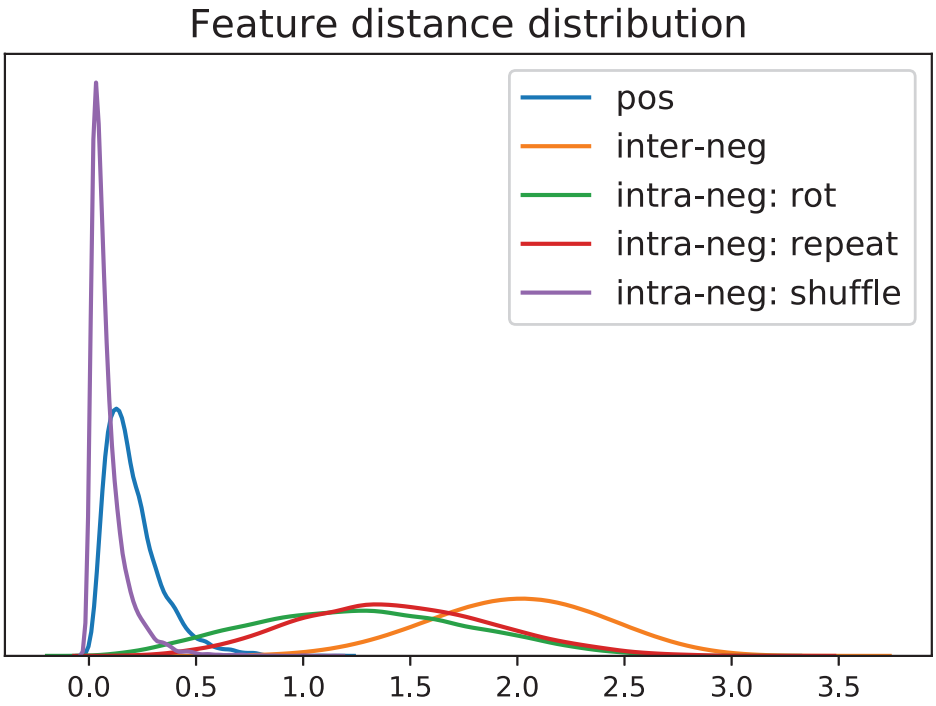


Figure 3.8: Feature distance distribution. The feature L2 distances are calculated using samples pairs from UCF101 split 1. For each sample pairs, one is the anchor, and the other one could be intra-positive, inter-negative, or intra-negative sample. The parameters of the network is randomly initialized without optimization. Curves are obtained using kernel density estimation (KDE).

even smaller than normal positive video clips. It is highly possible that these samples do not benefit contrastive learning targets during training. For frame repeating and frame shuffling, the overlap between them and inter-negative is larger, indicating that if treating them as positives, the optimization period is much harder. The best usage of intra-negative samples is to treat them (especially for frame repeating and clip rotation) as hard-negatives.

3.6.4 Best Option for Intra-Negative Samples

In Fig. 3.8, we find that for frame shuffling, the initial feature distances are smaller even than positive samples. Treating them as negatives makes it difficult to train the network. For frame repeating and clip rotation, their distributions are close, corresponding to the similar performances in both video retrieval (Table 3.5, Table 3.6 and Table 3.7) and recognition (Table 3.8 and Table 3.9) for most network backbones. Rotation is the best intra-negative generation function based on extensive experimental results. We think the reason lies in that intra-negative samples from frame

Table 3.11: Comparison with methods (i.e., BYOL and SimSiam) which do not need negative samples.

Method	Modality	Top1	Top5	Recognition
BYOL	RGB	15.8	32.6	77.9
BYOL	Res	16.4	25.8	24.4
SimSiam	RGB	43.9	57.0	64.0
SimSiam	Res	35.6	52.0	74.7
IICv2 (rotate)	Res	**53.1**	**70.1**	**78.6**

repeating are closer to inter-negatives, indicating that if treating rotation as negative samples, it can bring more sufficient temporal clues which are effective in feature discrimination.

When it comes to the situation that makes use of more than one intra-negative generation functions to get intra-negative samples, the training difficulty will greatly increase because frame shuffling and the other two transformations vary a lot according to Fig. 3.8. Though for the feature distance angle, it is similar between frame repeating and clip rotation, the useful discriminative features are various. Effective features for distinguishing the former one (generated by frame repeating) from anchor can detect whether there is movement or not. However, for rotated video clips, the action directions matter. These kinds of variety greatly increase the training difficulty. Therefore, we find that these intra-negative generation functions can contribute separately and rotation is the best intra-negative generation function among them.

3.6.5 Necessity of Negative Samples

Contrastive learning is used to optimize the network, and contrastive loss is proved to be a hardness-aware loss function in [138]. With intra-negative samples as hard negatives, the performance can be further enhanced. Though some image-based unsupervised learning works such as BYOL [39] and SimSiam [116] claimed that without negative samples, networks can also be trained and achieve good performance. With our findings, we should not deny the effectiveness of negative samples, especially our proposed intra-negative samples in self-supervised contrastive learning.

We also have small experiments which apply BYOL and SimSiam to videos with similar settings. As we can see from Table 3.11, our proposed methods can still

outperform these frameworks which do not need negative samples to train, showing that with proper hard-negative samples, good performances can also be obtained using a traditional contrastive learning scheme. The performance of BYOL in video retrieval is very low and some settings have severe overfitting problems (i.e., BYOL with residual modality, SimSiam with RGB modality) in action recognition, which requires careful choices in training settings in video self-supervised learning.

3.6.6 Limitations

One limitation for our IIC is that the quality of the trained model highly depends on the intra-negative generation functions. Clip rotation is proved to be the best option among our trials. However, clip rotation both changes the directions of movements as well as the spatial information. Though we have provided some analyses towards the potential explanation of the mechanism behind it, these proofs are not that solid. Another limitation is that it is special-designed for contrastive video representation learning, and only is compatible with methods that require negative samples in videos.

3.7 Conclusions

On the basis of IIC, we introduce many effective techniques and propose IIC, an improved inter-intra contrastive learning framework for self-supervised video representation learning. The advantages of intra- and inter-sample learning are combined by introducing intra-negative samples in contrastive learning. Three intra-negative sample generation functions are proposed which break the temporal relations in input video clips. Our framework is flexible and compatible with different settings such as different network backbones, different data modalities, as well as different intra-negative generation functions. Techniques such as strong data augmentations as well as the projection head are also applied to further enhance the performance. By using our framework, the trained models can extract better video representations when evaluated in two video tasks, video retrieval and video recognition. Extensive experiments validate the improvements brought by each part, as well as the general effectiveness using different network backbones. Discussions and visualizations validate that our IIC can capture better temporal clues and the potential mechanism. With only one model handling different inputs, we could surpass other methods by a large margin.

Chapter 4

Pretext-Contrastive Learning for Self-Supervised Video Representation

4.1 Introduction

With the development of convolutional neural networks (CNNs) and the help of many large-scale labeled datasets, the computer vision community has witnessed unprecedented success in many tasks such as object classification, detection, segmentation, and action recognition. For both image-level and video-level tasks, pre-training on larger datasets such as ImageNet [139] and Kinetics [45] is important to ensure satisfactory performance.

However, the world is abundant in images and videos, and annotating large-scale datasets requires a wealth of resources. In particular, the action recognition task generally requires properly trimmed action video clips to avoid unnecessary noise to ensure the performance, which makes the situation more serious. To leverage unlabeled data, many self-supervised learning methods have been proposed for efficient and versatile feature representation. These methods can be broadly divided into two categories, pretext task-based methods and contrastive learning methods.

Several tasks have been designed to constrain pretext task-based models to learn effective and informative representations. These tasks include solving jigsaw puzzles [93], image inpainting [94], and detecting image rotation angles [99]. For video data, some of these spatial tasks are also effective, together with temporal-related tasks such as predicting frame orders [23] or video clip orders [5, 24, 25], recognizing temporal transformations, and being sensitive to video playback speed [26–30]. A suitable combination [130] of such different tasks can help improve the performances

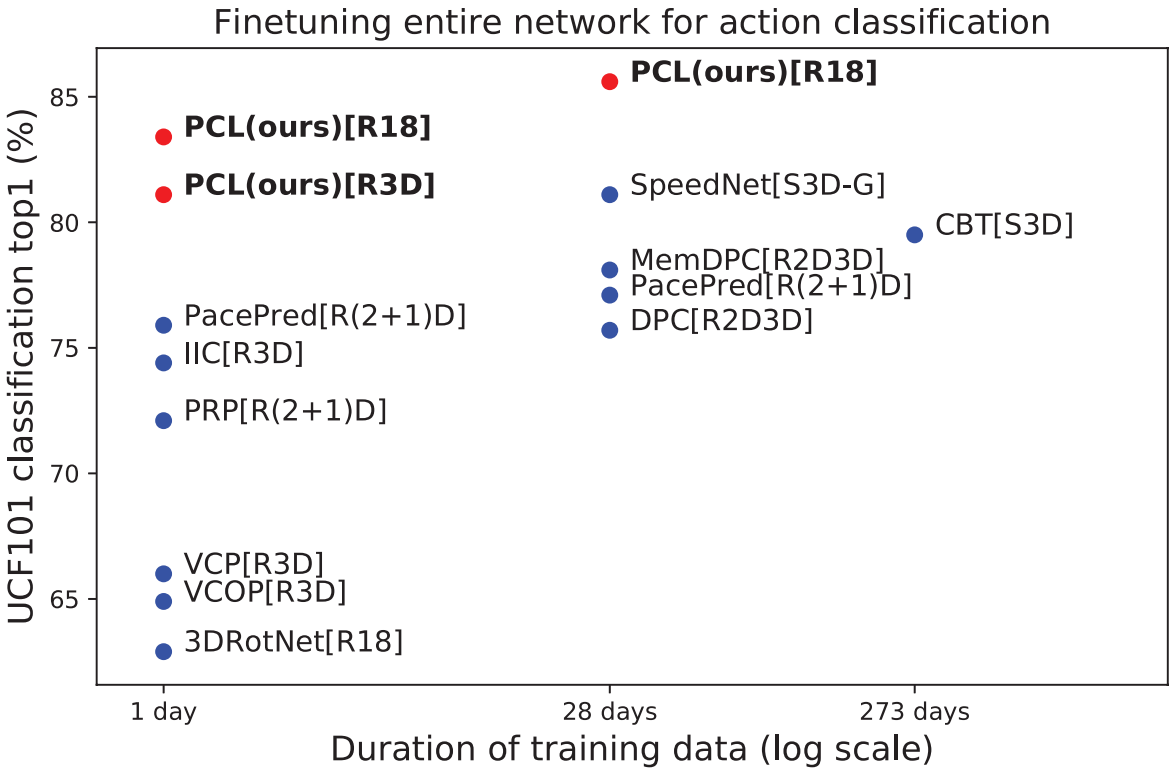


Figure 4.1: A glance at the performance of our proposals. Our results in this figure are based on one pretext task, VCP [3], the performance of which is only 66%. Results of other methods are from corresponding papers and results using the same input sizes ($16\times112\times112$) are used if provided, without using other data modalities such as optical flow, audio and text.

of the methods in video retrieval and recognition tasks. However, even though high accuracy can be achieved, it seems to be endless because there can be new and "better" pretext tasks. Identifying which pretext task is more effective and why it is are theoretically difficult to explain.

In contrastive learning methods [2, 32, 34, 35, 38], the solution is based on the comparison among different samples. The key idea is to distinguish one instance from another. Usually, different modalities and different spatial/temporal crops of the same video are treated as positives while samples from different videos are treated as negatives, even though they may belong to the same action category. Once the network can distinguish one instance from another, the learned features are discriminative and would be sufficient for downstream tasks such as video retrieval and recognition. Unlike those for pretext tasks, these methods may not consider sufficient temporal information because spatial features may be sufficient in some cases.

The combination of different pretext tasks and contrastive learning seems to be better than each on its own. Such kind of combination using one pretext task (pace prediction) has been firstly validated effective in video representation learning reported in a recent work [30]. However, the reason why the combination can be effective lacks discussion and the generality of this combination is unsure whether this phenomenon happens only for a specific pretext task or not.

In this chapter, based on the success of pretext tasks and contrastive learning, we want to explore what kind of combination can boost both. We propose **P**retext-**C**ontrastive **L**earning (PCL), which also facilitates the advantages of some data processing strategies such as residual clips (Sec. 2) and strong data augmentations [38]. With PCL, huge improvements over the corresponding baselines can be achieved, as shown in Fig. 4.1. Better performance can be obtained over recent works while even using much smaller (around 3.6%) data for pre-training. We should clarify that we are not proposing new pretext tasks, or contrastive learning methods; instead, we want to bridge the gaps between pretext tasks and contrastive learning with comprehensive experimental investigation and discussion to find the best strategy facilitating the advantages of these technologies. And this work is trying to set new baselines in self-supervised learning in videos.

To prove the effectiveness of our PCL, three pretext task-based methods are set as baselines, together with the contrastive learning method. Different network backbones are tested to eliminate biases. Experimental results prove the effectiveness and the generality of our proposal. The proposed PCL is closer to a framework or a strategy rather than a simple method as it is flexible and can be applied to many existing solutions. And we have lifted benchmarks to a new level by tiny changes, setting new baselines in self-supervised video representation learning.

The contributions of this work can be summarized as:

- We propose a joint optimization framework, utilizing the advantage of both pretext tasks and contrastive learning, together with proper training settings.

- Experiments demonstrate that huge improvements can be obtained by using our proposal, and we can also achieve state-of-the-art performances in two evaluation tasks on two benchmark datasets.

- Our proposal is validated based on three pretext task baselines and different network backbones, showing the effectiveness and the generality of our PCL.

- Analysis shows some connections between pretext tasks and contrastive learning, helping to understand the potential mechanism behind the simple combination.

4.2 Related Works

In this section, we divide the existing self-supervised learning methods into two categories according to their optimization targets: pretext tasks and contrastive learning. Because no labels are available for self-supervised learning, pretext tasks based methods will set special tasks as the training target, such as detecting transformations. Contrastive learning-based methods will use positive pairs and negative pairs to train the network, and the generation of these sample pairs is based on the sample indexes.

4.2.1 Pretext Tasks

Self-supervised learning methods were first proposed for images. Spatial pretext tasks include solving jigsaw puzzles [93], detecting image rotations [99], image channel prediction [95], and image inpainting [94]. Prior works also include image reconstruction using autoencoders [97] and variational autoencoders [98].

For video data, some image-based pretext tasks can be directly applied or extended, such as detecting rotation angles [4] and completing space-time cubic puzzles [103]. Compared to image data, videos have an additional temporal dimension. Therefore, to utilize temporal information, many works have designed temporal-specific tasks. In [23], the network was trained to distinguish whether the input frames were in the correct order. [24] trained their odd-one-out network (O3N) to identify unrelated or odd video clips. The order prediction network (OPN) [25] was trained by predicting the correct order of shuffled frames. The video clip order prediction network [5] used video clips together with a spatio-temporal CNN during training. Further, [3] utilized spatial and temporal transformations to train the network. Many recent works [26–30] have started to utilize the playback speed of the input video clips. SpeedNet [26] was trained to detect whether a video is playing at a normal rate or a sped-up rate. [27] trained a network to sort video clips according to the corresponding playback rates. The playback rate perception (PRP) [28] used an additional reconstructing decoder branch to help train the model. [29] and [30] also utilized additional transformations to help train the model.

All these pretext tasks can be set as the main branch and can be combined with our PCL for better performance.

4.2.2 Contrastive Learning

The success of contrastive learning also originated from image tasks [140]. The key idea of contrastive learning is to minimize the distance within positive pairs in the feature space while maximizing the distance between negative pairs. After contrastive loss was proposed [31], contrastive learning has become the mainstream method for self-supervised learning of image data. Contrastive predictive coding (CPC) [32] attempted to learn the future from the past by using sequential data. Deep InfoMax [33] and Instance Discrimination [34] were proposed to maximize information probability from the same sample. Contrastive multiview coding (CMC) [2] used different views (e.g. different color spaces) from the same sample. Momentum Contrast (MoCo) [35, 36] used a momentum-updated encoder to conduct contrastive learning. In SimCLR [38], different combinations of data augmentation methods were tested for paired samples. Bootstrap Your Own Latent (BYOL) [39] trained the network without negative samples.

The above-mentioned methods mainly focus on image data. Some technologies have been successfully applied to video data. The concept of CMC can be easily adapted to videos by simply using video data as the model input. Similar to CPC, DPC [43] and MemDPC [37] were proposed to handle video data. In Sec. 3, we have introduced intra-negative video samples to enhance temporal representation for contrastive learning. These methods are all based on visual data only. The contrastive learning concept can be extended to additional modalities of video, such as audio [40, 41], text, and descriptive data [42].

Most of these contrastive learning methods utilize a noise contrastive estimation (NCE) loss [127] for robust and effective training. Wang et al. [141] explored the learned features and proposed a new loss function, align-uniform loss, which is a possible substitute for the NCE loss. In our PCL, we used the NCE loss for optimization. Other contrastive loss functions are also compatible with our framework.

4.2.3 Methods Combinations

A combination of several pretext tasks with proper weights can yield better performances [130] than when they are used alone. Many existing pretext task-based

methods are beyond one simple pretext task and are already a combination of some particular tasks. We have listed many pretext tasks, and the potential combinations among them are extensive. These pretext tasks vary widely, and determining why one pretext task or one combination is better than another is difficult.

The combination of pretext tasks and contrastive learning has been attempted in a recent work [30]. However, except for the reported results, few analyses have been conducted and the combination may be only effective on a specific task. In this section, we address this issue and show the generality of the combination of pretext task and contrastive learning as it can boost the performance of both. Improvements over three pretext task baselines also reveal that the effective settings can be generalized to a lot of pretext tasks.

4.3 Methodology

4.3.1 Motivation

Pretext task methods and contrastive learning methods can have good performances on their own. And some questions arise automatically. 1) Can a simple combination of a pretext task-based method and a contrastive learning method boost each other and achieve better performance? 2) Will it be effective only for a specific pretext task, or general enough for many pretext tasks?

4.3.2 PCL: Pretext-Contrastive Learning

The goal for self-supervised video representation learning is to learn effective feature representations from videos using a backbone network f_θ. The commonly used networks are based on spatio-temporal convolutions, where the input video v_i is decoded to a sequence of frames and several frames are stacked to form video clips x_{v_i}. Video features can be generated by using $f_\theta(x)$.

Pretext Task

For pretext task-based methods, one or several tasks are used to train the network in a supervised manner. Most pretext tasks are classification tasks. For example, VCP [3] used different transformations on the input video clip x and trained the network by distinguishing which transformation was conducted. 3DRotNet [4] was trained by detecting the rotation angles of the input clip. VCOP [5] shuffled video

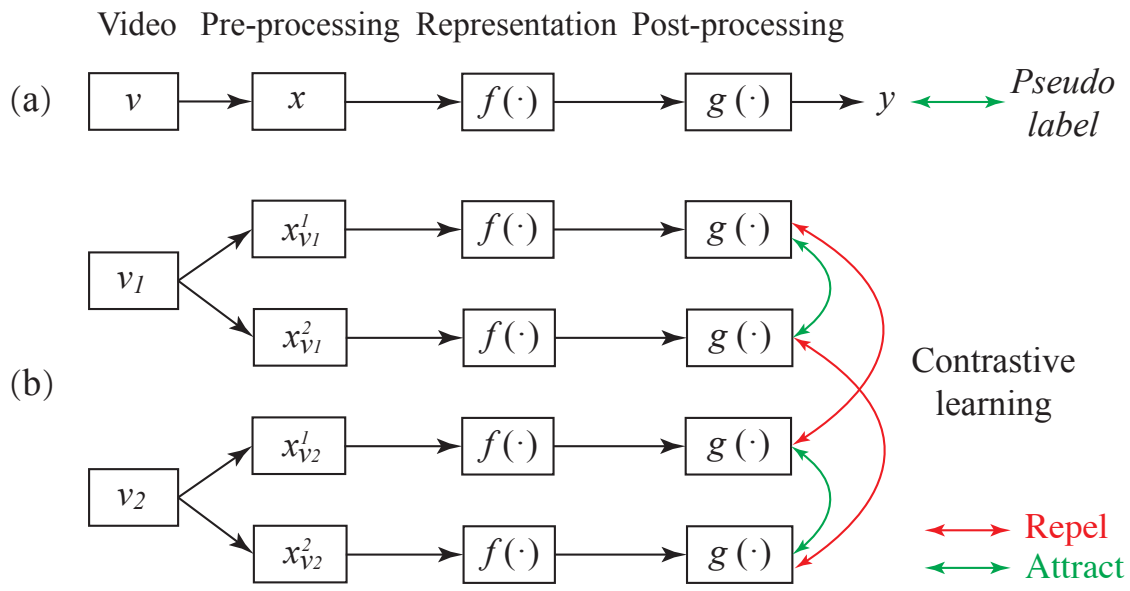


Figure 4.2: (a) Learning scheme of pretext tasks; (b) Learning scheme of contrastive learning methods).

clips and trained the network by predicting the correct order class of the inputs. All these pretext tasks can be concluded as designing a proper classification task. The video clip x needs to be transformed by a specific transformation function $t(x, y)$, where y is the label of the corresponding transformation. Then the optimization target of these pretext tasks becomes

$$\underset{\forall v_i}{\text{minimize}}\ \mathcal{L}_{cls}(g(f_\theta(t(x_{v_i}, y))), y), \tag{4.1}$$

where $g(\cdot)$ is the post-process network to process extracted features and $\mathcal{L}_{cls}$ is usually set as cross-entropy loss because the corresponding pretext tasks usually belong to classification tasks. The learning scheme of pretext tasks is illustrated in Fig. 4.2 (a).

Contrastive Learning

For contrastive learning methods, after extracting features from the backbone, a two-linear-layer multi-layer perceptron (MLP) is usually used to project features $f_\theta(x)$ to another feature space. Let us denote the projector network as $h(\cdot)$. Positive pairs and negative pairs are required to constrain the network. $x^1_{v_i}$ is one video clip from the video v_i, and when another video clip $x^2_{v_i}$ is from the same video, these two video clips are treated as a positive pair. Conversely, when a video clip x_{v_j} is from

a different video, v_j. Then x_{v_i} and x_{v_j} are a negative pair. The encoded feature in the projected feature space is $h(f(x_{v_i}))$, which is denoted as z_{v_i} for simplicity. Let us define $D(z_{v_i}, z_{v_j})$ as the similarity distance between feature z_{v_i} and z_{v_j}; then for video v_i, the contrastive learning target is

$$\text{minimize } \mathcal{L}_{NCE}^{v_i} = \mathcal{L}_{NCE}^{v_i^1} + \mathcal{L}_{NCE}^{v_i^2}, \tag{4.2}$$

where

$$\begin{aligned} \mathcal{L}_{NCE}^{v_i^1} &= -\log \frac{D(z_{v_i}^1, z_{v_i}^2)}{D(z_{v_i}^1, z_{v_i}^2) + \sum_{j \neq i} D(z_{v_i}^1, z_{v_j}^1)}, \\ \mathcal{L}_{NCE}^{v_i^2} &= -\log \frac{D(z_{v_i}^1, z_{v_i}^2)}{D(z_{v_i}^1, z_{v_i}^2) + \sum_{j \neq i} D(z_{v_i}^2, z_{v_j}^2)}. \end{aligned} \tag{4.3}$$

In practice, video features (i.e., z_{v_i}) are normalized in the feature space and the similarity distance $D(z_{v_i}, z_{v_j})$ is calculated by the inner product. In contrastive learning, instances with different indexes can be treated as negative samples and at most $N-1$ negative samples can be used, where N is the size of the dataset. To accelerate training, memory bank [34] technologies are adopted to save extracted features from previous epochs and k negative samples are sampled from the corresponding memory banks. This procedure is similar to [2, 91]. This kind of learning scheme is illustrated in Fig. 4.2 (b).

Joint Optimization Framework

As we can see from the optimization targets of pretext tasks (Eq. 4.1) and contrastive learning (Eq. 4.2 and Eq. 4.3), pretext task-based methods focus more within the sample while contrastive learning methods try to distinguish one instance from another. A combination of them may take the advantage of both, ensuring the network to have a local-global view.

There are several pretext tasks, and some tasks use only one video clip to conduct experiments such as 3DRotNet [4], which rotated the input video clip and trained the model by predicting the rotation angles. Some tasks use multiple video clips during training, such as VCOP [5], which shuffled the temporal order of several video clips, and VCP [3], which utilized spatial and temporal transformations. The training styles for almost all pretext tasks can also be divided into two categories, single-clip methods and multi-clip methods. For a better understanding of these three baselines, we illustrate the training scheme in Fig. 4.3.

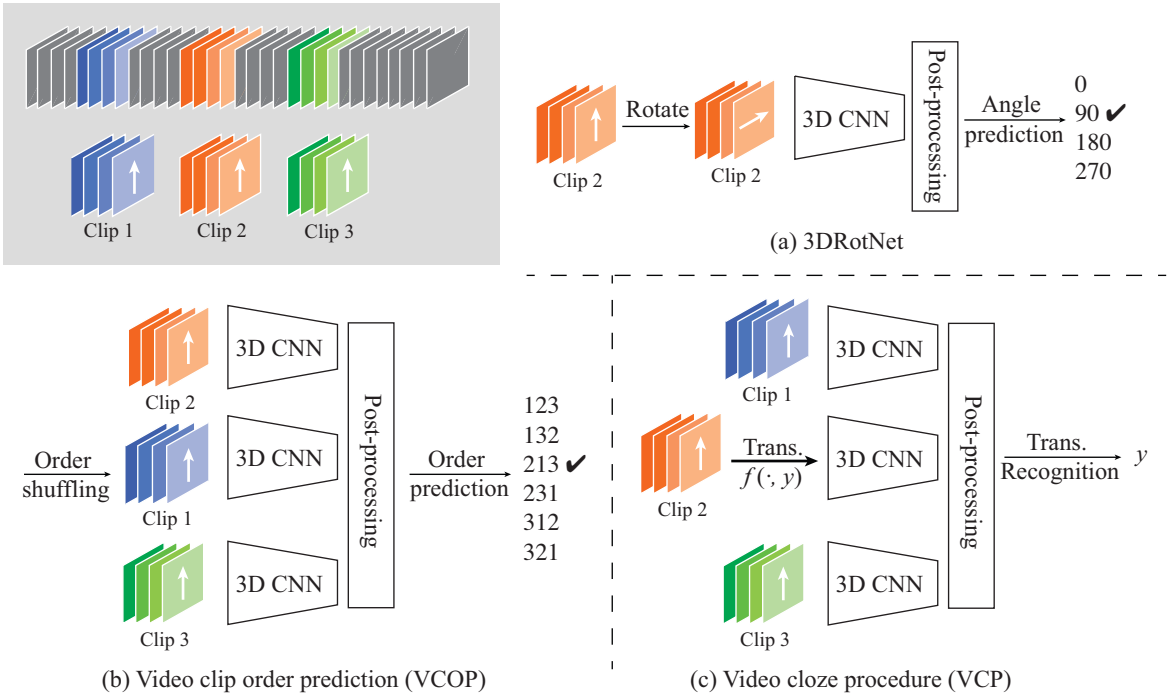


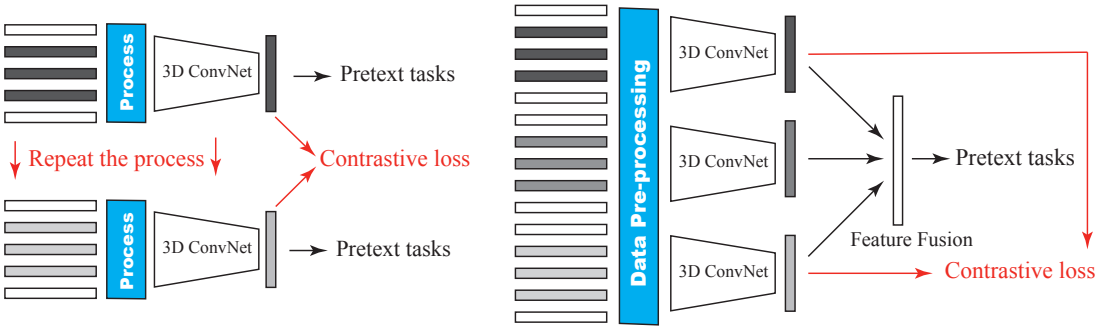
Figure 4.3: The overview of three pretext task baselines, 3DRotNet[4], VCOP [5], and VCP [3]. These three methods cover a variety settings of existing pretext tasks. In VCP, the transformation includes several spatial-related and temporal related tasks.

In this work, we choose 3DRotNet, VCOP, and VCP as our pretext task baselines because of the variance among these three methods. We show some features of these pretext tasks in Table 4.1. The situation can cover almost all existing video pretext tasks based on these points of view.

We illustrate the use of our proposal in Fig. 4.4. For single-clip methods, the contrastive loss will use the encoded features from the backbone network. As contrastive loss requires a positive pair and negative pairs to train, the encoding process is duplicated. The input video clip is generated from the same video as the original path, which can be treated as a positive pair. Negative pairs are taken

Table 4.1: Variety of the chosen pretext tasks. "trans." is short for the word transformation.

Pretext task	Spatial trans.	Temporal trans.	Clip settings
3DRotNet [4]	✓		Single
VCOP [5]		✓	Multiple
VCP [3]	✓	✓	Multiple



(a) Pretext-Contrastive Learning in single-clip methods

(b) Pretext-Contrastive Learning in multi-clip methods

Figure 4.4: The use of PCL in pretext task-based methods. (a) For single-clip methods, two different clips from the same video will be processed and the contrastive loss will be calculated among one batch of data. (b) For multi-clip methods, different clips from the same video have been already processed and the contrastive loss can be easily calculated. The data pre-processing procedure includes strong data augmentation transformations and converting to residual clips.

directly from one batch of data because different samples are from different videos in one training batch.

For multi-clip methods, different video clips are set as inputs and they are encoded to features using a shared encoder. These features are natural positive pairs because they are from the same video. Negative pairs are also from video clips from one batch of data.

It can be observed that it is very simple to construct a joint optimization framework based on any pretext task baseline method, and the final training loss becomes

$$\mathcal{L}_{total} = \mathcal{L}_{pretext} + \alpha \mathcal{L}_{contrast}, \tag{4.4}$$

where α is a weight to balance losses between pretext tasks and contrastive learning. For $\mathcal{L}_{pretext}$ and $\mathcal{L}_{contrast}$, they came from Eq. 4.1 and Eq. 4.3. For convenience, we rewrite them here.

$$\mathcal{L}_{pretext} = \mathcal{L}_{cls}(g(f_\theta(t(x_{v_i}, y))), y), \tag{4.5}$$

$$\mathcal{L}_{contrast} = \mathcal{L}_{NCE}^{v_i^1} + \mathcal{L}_{NCE}^{v_i^2} \tag{4.6}$$

4.3.3 Data Processing Strategies

To further boost the performance, we mainly introduce two different kinds of processing strategies on data, namely residual clips and augmentation transformations.

Residual clips. Most video-based self-supervised learning methods use 3D convolutional networks to process data, and the corresponding input is video clips, which are stacked RGB frames. Residual clips have been introduced in Sec. 2, showing that they can function well with several pretext tasks. We introduce residual clips here to further show its effectiveness on different methods in self-supervised learning.

Here we use $frame_i$ to represent the i_{th} frame data, and $Frame_{i\sim j}$ denotes the stacked frames from the i_{th} frame to the j_{th} frame. The process to get residual frames can be formulated as follows,

$$ResClip = Frame_{i+1\sim j+1} - Frame_{i\sim j}, \tag{4.7}$$

where $Frame_{i+1\sim j+1}$ can be easily obtained by shifting frames along the temporal axis in video clips. The *ResClip* here will be then directly fed into the network for feature extraction.

Augmentation transformations. It is widely acknowledged that data augmentation methods enhance performance in most cases. However, in previous methods in video-based self-supervised learning, only a few data augmentations were conducted such as random cropping in the spatial axis and temporal jittering. However, some recent works [35, 39] started to use strong augmentations in images, such as color distortion and Gaussian blur, and have achieved improvements over the corresponding baselines.

Though these data augmentations are conducted on images, we adopted this kind of processing and applied it to video frames. The motivation is that motion features should be similar even though frames are blurred or distorted by color. Another motivation is that we wonder whether it will also boost the performance of residual clips because there will be much less color information in residual clips.

4.4 Experiments

To demonstrate the effectiveness of the proposed PCL framework, we used 1) three pretext task baselines with traditional contrastive learning baseline method, 2) four

different backbone networks including different convolution types, and 3) two evaluation tasks (i.e., video retrieval and recognition) in our experiments.

4.4.1 Data Preparation

We mainly used two benchmark datasets, UCF101 [56] and HMDB51 [57] in our experiments, as our baselines did [4, 5, 3]. The UCF101 dataset consists of 13,320 videos in 101 action categories. HMDB51 is comprised of 7,000 videos with a total of 51 action classes. The official splits only contain training sets and testing sets. We randomly selected 800 and 400 videos from the training splits for UCF101 and HMDB51 datasets, respectively, to form the validation set. The best performance on the validation set will be saved and evaluated in video retrieval and recognition. To further evaluate the effectiveness of our PCL, we also utilized Kinetics-400 dataset [9] to train. Kinetics-400 consists 400 action classes and contains around 240k videos for training, 20k videos for validation, and 40k videos for testing. Kinetics-400 dataset is only used in the pre-training process.

Because spatio-temporal convolutions were used to train our models, we followed [6] and resized videos in size 128×171. Sixteen successive frames are sampled to form a video clip. Random spatial/temporal cropping was conducted to generate an input video clip of size $16 \times 112 \times 112$, where the channel number 3 was ignored. In addition to random cropping, other augmentation transformations we used in our experiments include random color jittering, randomly converting to grayscale, Gaussian blur, and random flipping.

4.4.2 Baselines

Because our PCL is a combination of pretext tasks and contrastive learning, the baselines should be set as the pretext task or contrastive learning.

There are several pretext task-based methods in self-supervised video representation learning. We chose three works: 3DRotNet [4], VCOP [5], and VCP [3]. 3DRotNet is trained by recognizing the rotated angles (degrees are from [0, 90, 180, 270]) of the input video clip. VCOP aims to detect the correct orders of several input video clips. For example, for three video clips with original order "1, 2, 3", there are totally six possible options after order shuffling, such as "1, 2, 3", "2, 3, 1", "3, 2, 1". VCP conducts different types of transformations and the network is trained to distinguish which transformation has been performed. These pretext tasks, as well as the training styles, are different. For example, 3DRotNet is a one-clip method

while VCOP and VCP use several video clips as input data. The other reason is that these three pretext tasks are from three different categories. 3DRotNet uses rotation, which is more related to spatial information. VCOP cares about temporal orders of input clips, which only uses temporal information. The processing which VCP chooses from is a mixture of temporal and spatial transformations. There exist many pretext tasks in video-based self-supervised learning and we cannot conduct all experiments. However, other pretext tasks can be easily classified into one of these three categories and we think the effectiveness of our proposal on these three pretext tasks can prove the generality of our PCL.

Contrastive learning is widely used in image-based self-supervised learning and has been explored in videos in [91, 37, 43]. For a fair comparison, our contrastive learning baseline will use the same framework as our PCL while the network will be optimized only by $\mathcal{L}_{contrast}$ in Eq. 4.4, without using $\mathcal{L}_{pretext}$.

4.4.3 Network Backbones

For the network backbone, there are several 3D CNNs such as C3D [6], R3D, ResNet-18-3D [8], and R(2+1)D [7]. Different network backbones were used in our experiments to eliminate model biases. R3D and ResNet-18-3D are composed of 3D convolution instead of 2D convolution in the original ResNet [60] while the numbers of convolutional layers in each residual block vary. To compare with the baselines, we used the same network architectures as them. It is possible to use other network architectures such as I3D [9], S3D [10], or other deeper networks, but we simply follow the baselines for fair comparisons.

A two-linear-layer multi-layer perceptron (MLP) is used to process features from the same backbone. Therefore, this part can be treated as the post-processing for the contrastive learning part, paralleling with the post-processing of pretext tasks. The MLP is in an *fc-relu-fc* style. After projection, feature dimensions are reduced to 128 in our experiments.

4.4.4 Evaluation Tasks

To evaluate the performance of the trained models, two evaluation tasks were used: video retrieval and video recognition. After self-supervised training, the trained models can be evaluated directly in video retrieval tasks on both UCF101 and HMDB51. Note that the self-supervised learning part was only conducted on UCF101 *split* 1. Therefore, when conducting video retrieval on UCF101, the task-level

generalization ability was tested. When conducting video retrieval on HMDB51 using the same model, both task-level and dataset-level generalization abilities were tested.

Video retrieval is conducted based on video-level features. 3D ConvNets can extract features from video clips, and features of video clips are averaged if they are from the same video. Thus, video-level features can be generated and the k-nearest neighbors (kNN) algorithm is used to check whether the retrieved video has the same action category as the query video.

Action recognition is a fundamental task in video representation learning. Following previous works, we also conducted experiments by fine-tuning trained models on both UCF101 and HMDB51 datasets to check the transfer learning ability of the models.

4.4.5 Experimental Details

In all of our experiments, the batch size is set to 16 and the training lasts for 200 epochs. The initial learning rate is 0.01 for self-supervised learning. Models with the best performance on the validation datasets are saved then used to test the performance in the video retrieval task. For video recognition tasks, the same models are fine-tuned for 150 epochs and the initial learning rate is set to 0.001. The best performance on the validation dataset is evaluated on the corresponding test splits. Stochastic Gradient Descent (SGD) is used for optimization for both training periods. The hyper-parameter in Eq. 4.4, α is set to 0.5 to balance pretext task loss and contrastive loss.

4.5 Results and Analyses

In this section, we first compare our proposed method with baseline methods. To further prove the effectiveness of our PCL framework, we also compare our results with current state-of-the-art methods. We mainly used VCP as the baseline pretext task and used C3D, R3D, or R(2+1)D as the network backbone. For the other two methods, 3DRotNet and VCOP, we used the same mainstream backbones reported in the corresponding papers: ResNet-18-3D for 3DRotNet and C3D for VCOP.

Table 4.2: Comparisons with baselines on *split* 1 of UCF101. Best results in each block are in **bold**.

Method	Backbone	Video retrieval					Recog.
		Top1	Top5	Top10	Top20	Top50	
VCP (baseline) [3]	C3D	17.3	31.5	42.0	52.6	67.7	68.5
Contrastive only [2]	C3D	38.9	56.9	65.7	74.4	84.3	78.0
PCL	C3D	**50.3**	**67.3**	**75.7**	**83.4**	**91.2**	**79.8**
VCP (baseline) [3]	R3D	18.6	33.6	42.5	53.5	68.1	66.0
Contrastive only [2]	R3D	44.7	62.4	71.6	79.6	88.8	79.3
PCL	R3D	**48.1**	**64.7**	**73.9**	**82.0**	**90.6**	**79.9**
VCP (baseline) [3]	R(2+1)D	19.9	33.7	42.0	50.5	64.4	66.3
PCL	R(2+1)D	**42.8**	**59.9**	**69.5**	**78.0**	**87.6**	**79.9**
3DRotNet (baseline) [4]	R3D-18	14.2	25.2	33.5	43.7	59.5	62.9
PCL	R3D-18	**33.7**	**53.5**	**64.1**	**73.4**	**85.0**	**81.5**
VCOP (baseline) [5]	C3D	12.5	29.0	39.0	50.6	66.9	65.6
PCL	C3D	**39.0**	**59.1**	**67.5**	**76.8**	**87.4**	**79.2**

4.5.1 Comparison with Baselines

All models were pre-trained on UCF101 *split* 1 and tested on both UCF101 and HMDB51 datasets. Results are presented in Table 4.2 and Table 4.3, respectively.

For the pretext task VCP with the C3D backbone, the baseline is only 17.3% in video retrieval and 68.5% in recognition on the UCF101 dataset. When maintaining the main training architecture and using contrastive loss only, the performance can reach 38.9% in retrieval and 78.0% in video recognition. This performance is much higher than the pretext task. One reason is that it already benefits from our data processing strategies. Our PCL yielded 50.3% at top 1 retrieval accuracy on the UCF101 dataset, which is 33.0% points above the C3D baseline for pretext task and also 11.4% points higher than our strong contrastive learning baseline. In video recognition, our PCL can also yield the best performance.

Similar results can be found when we used different network backbones on the basis of VCP. Our PCL can achieve more than double the performance of the corresponding baseline at top 1 retrieval accuracy and over 10% points improvement when we use R3D and R(2+1)D as the network backbone. These results show that our PCL can boost the performance of both VCP and contrastive learning.

Table 4.3: Comparisons with baselines. Results are evaluated on *split* 1 of HMDB51. Best results in each block are in **bold**.

Method	Backbone	Video retrieval					Recog.
		Top1	Top5	Top10	Top20	Top50	
VCP (baseline) [3]	C3D	7.8	23.8	35.3	49.3	71.6	32.5
Contrastive only	C3D	15.1	34.9	47.2	61.5	82.1	45.5
PCL	C3D	**19.6**	**41.5**	**44.8**	**70.2**	**85.9**	**46.1**
VCP (baseline) [3]	R3D	7.6	24.4	36.3	53.6	76.4	31.5
Contrastive only	R3D	17.3	38.6	51.2	65.3	83.4	**46.3**
PCL	R3D	**19.2**	**42.0**	**55.3**	**69.1**	**86.7**	46.1
VCP (baseline) [3]	R(2+1)D	6.7	21.3	32.7	49.2	73.3	32.2
PCL	R(2+1)D	**19.6**	**41.1**	**56.2**	**71.1**	**86.5**	**45.9**
3DRotNet (baseline) [4]	R3D-18	6.2	18.7	31.0	46.6	70.5	33.7
PCL	R3D-18	**12.4**	**34.4**	**48.4**	**65.4**	**83.6**	**47.4**
VCOP (baseline) [5]	C3D	7.4	22.6	34.4	48.5	70.1	28.4
PCL	C3D	**14.9**	**35.9**	**48.9**	**63.6**	**82.8**	**42.2**

When we look at other pretext baselines in Table 4.2 and Table 4.3, the same trend can be found. Our PCL can outperform the corresponding pretext task baselines, 3DRotNet and VCOP, by a large margin. These results reveal that the effectiveness of our PCL is not limited to only one pretext task, but general enough to other methods. Also, we want to mention that VCP cares much about spatial and temporal transformations, VCOP uses temporal information only and 3DRotNet uses rotation which is much more related to spatial information. The effectiveness of PCL on these three baselines reveals the potential that our PCL can boost the performance of other existing pretext task-based methods in self-supervised video representation learning.

4.5.2 Comparison with State-of-the-art Methods

There are too many pretext tasks in video self-supervised learning and it is impossible for us to embed our proposal to all these methods. The baselines we used in our study are not currently state-of-the-art methods. Some very recent works have used new pretext tasks such as pace prediction [30] or more complex temporal transformation recognition [29] and achieved better performances. Here we compared our methods

Table 4.4: Comparison with state-of-the-art methods in video retrieval on UCF101. Most results are from the corresponding papers.

Methods	Backbone	Top1	Top5	Top10	Top20	Top50
MemDPC [37]	R2D3D	20.2	40.4	52.4	64.7	-
MemDPC-Flow [37]	R2D3D	**40.2**	**63.2**	**71.9**	**78.6**	-
PRP [28]	C3D	23.2	38.1	46.0	55.7	68.4
PacePred [30]	C3D	31.9	49.7	59.2	68.9	80.2
IIC	C3D	**55.1**	**72.1**	**78.4**	**84.0**	**91.6**
PCL (VCOP)	C3D	39.0	59.1	67.5	76.8	87.4
PCL (VCP)	C3D	50.3	67.3	75.7	83.4	91.2
PRP [28]	R3D	22.8	38.5	46.7	55.2	69.1
IIC	R3D	**53.1**	**70.1**	**77.4**	**84.0**	**91.4**
PCL (VCOP)	R3D	38.9	57.8	66.6	76.1	86.0
PCL (VCP)	R3D	48.1	64.7	73.9	82.0	90.6
PRP [28]	R(2+1)D	20.3	34.0	41.9	51.7	64.2
PacePred [30]	R(2+1)D	25.6	42.7	51.3	61.3	74.0
IIC	R(2+1)D	**50.6**	**68.3**	**76.0**	**82.9**	**90.5**
PCL (VCOP)	R(2+1)D	16.6	33.3	43.1	55.5	72.6
PCL (VCP)	R(2+1)D	42.8	59.9	69.5	78.0	87.6
RTT [29]	R3D-18	26.1	48.5	59.1	69.6	82.8
PacePred [30]	R3D-18	23.8	38.1	46.4	56.6	69.8
IIC	R3D-18	**56.2**	**71.3**	**77.5**	**84.6**	**91.6**
PCL (3DRotNet)	R3D-18	33.7	53.5	64.1	73.4	85.0
PCL (VCP)	R3D-18	55.1	71.2	78.9	85.5	92.3

with state-of-the-art methods to demonstrate the effectiveness of our PCL. We want to clarify that there are some other works that used larger pre-trained datasets together with audio or text information of videos and achieved even higher performance [40–42]. Here, we did not include them and only referred to these methods using similar settings for fair comparisons.

The results for video retrieval in UCF101 and HMDB51 datasets are shown in Table 4.4 and Table 4.5, respectively. Note that we have proved the improvements over the corresponding baselines in Table 4.2 and Table 4.3, we do not include them in these tables. From these tables, we can see that by using the proposed PCL, we can easily outperform other state-of-the-art methods except for our IIC (Sec. 3), no matter which backbone is used. Here we use the best settings of IIC, which uses rotation as an intra-negative generation function. The solution of PacePred [30]

Table 4.5: Comparison with state-of-the-art methods in video retrieval on HMDB51. Most results are from the corresponding papers.

Methods	Backbone	Top1	Top5	Top10	Top20	Top50
MemDPC [37]	R2D3D	7.7	25.7	40.6	57.7	-
MemDPC-Flow [37]	R2D3D	**15.6**	**37.6**	**52.0**	**65.3**	-
PRP [28]	C3D	10.5	27.2	40.4	56.2	75.9
PacePred [30]	C3D	12.5	32.2	45.4	61.0	80.7
IIC	C3D	19.5	**44.7**	**58.7**	**73.1**	**89.3**
PCL (VCOP)	C3D	14.9	35.9	48.9	63.6	82.8
PCL (VCP)	C3D	**19.6**	41.5	44.8	70.2	85.9
PRP [28]	R3D	8.2	25.8	38.5	63.3	75.9
IIC	R3D	**20.4**	**43.1**	**56.3**	**70.5**	86.3
PCL (VCOP)	R3D	14.3	34.0	48.3	62.1	81.9
PCL (VCP)	R3D	19.2	42.0	55.3	69.1	**86.7**
PRP [28]	R(2+1)D	8.2	25.3	36.2	51.0	73.0
PacePred [30]	R(2+1)D	12.9	31.6	43.2	58.0	77.1
IIC	R(2+1)D	**20.0**	**43.4**	56.0	70.3	**86.5**
PCL (VCOP)	R(2+1)D	7.9	23.8	35.9	51.0	74.7
PCL (VCP)	R(2+1)D	19.6	41.1	**56.2**	**71.1**	**86.5**
PacePred [30]	R3D-18	9.6	26.9	41.1	56.1	76.5
IIC	R3D-18	**20.7**	**45.0**	57.6	71.6	86.1
PCL (3DRotNet)	R3D-18	12.4	34.4	48.4	65.4	83.6
PCL (VCP)	R3D-18	20.2	43.6	**59.1**	**72.5**	**86.6**

is already a combination of one pretext task (i.e., video speed recognition) and contrastive learning. We can still outperform their results by a large margin based on three network backbones. The best top-1 video retrieval performance in the UCF101 dataset is 55.1%, achieved by our PCL using ResNet-18-3D network backbone and the corresponding pretext task is VCP [3]. Similar trend can be found in HMDB51 dataset in Table 4.5. We can lift the corresponding pretext baselines by a large margin.

The results for video recognition are shown in Table 4.6. We can observe that without our proposal, the performances of the corresponding baseline methods are lower than those of recent state-of-the-art methods. However, with the proposed PCL, which only has minor changes in the baselines, the performances can be significantly improved. In most settings, PCL performs better than state-of-the-art methods. From this table, we can also see that the settings of existing methods vary from one to another, such as using different sizes of input data, different

Table 4.6: Comparisons with the state-of-the-art self-supervised methods.

Method	Date	Pre-train	ClipSize	Network	UCF	HMDB
OPN [25]	2017	UCF	227^2	VGG	59.6	23.8
DPC [43]	2019	K400	16×224^2	R3D-34	75.7	35.7
CBT [42]	2019	K600+	16×112^2	S3D	79.5	44.6
SpeedNet [26]	2020	K400	64×224^2	S3D-G	81.1	48.8
MemDPC [37]	2020	K400	40×224^2	R-2D3D	78.1	41.2
VCOP [5]	2019	UCF	16×112^2	C3D	65.6	28.4
VCP [3]	2020	UCF	16×112^2	C3D	68.5	32.5
PRP [28]	2020	UCF	16×112^2	C3D	69.1	34.5
RTT [29]	2020	K400	16×112^2	C3D	69.9	39.6
PCL (VCOP)		UCF	16×112^2	C3D	79.8	41.8
PCL (VCP)		UCF	16×112^2	C3D	**81.4**	**45.2**
VCOP [5]	2019	UCF	16×112^2	R3D	64.9	29.5
VCP [3]	2020	UCF	16×112^2	R3D	66.0	31.5
PRP [28]	2020	UCF	16×112^2	R3D	66.5	29.7
IIC	2020	UCF	16×112^2	R3D	78.6	43.4
PCL (VCOP)		UCF	16×112^2	R3D	78.2	40.5
PCL (VCP)		UCF	16×112^2	R3D	**81.1**	**45.0**
VCOP [5]	2019	UCF	16×112^2	R(2+1)D	72.4	30.9
VCP [3]	2020	UCF	16×112^2	R(2+1)D	66.3	32.2
PRP [28]	2020	UCF	16×112^2	R(2+1)D	72.1	35.0
RTT [29]	2020	UCF	16×112^2	R(2+1)D	81.6	46.4
PacePred [30]	2020	UCF	16×112^2	R(2+1)D	75.9	35.9
PacePred [30]	2020	K400	16×112^2	R(2+1)D	77.1	36.6
PCL (VCOP)		UCF	16×112^2	R(2+1)D	79.2	41.6
PCL (VCP)		UCF	16×112^2	R(2+1)D	79.9	45.6
PCL (VCP)		K400	16×112^2	R(2+1)D	**85.7**	**47.4**
3D-RotNet [4]	2018	K400	16×112^2	R3D-18	62.9	33.7
ST-Puzzle [103]	2019	K400	16×112^2	R3D-18	65.8	33.7
DPC [43]	2019	K400	16×128^2	R3D-18	68.2	34.5
RTT [29]	2020	UCF	16×112^2	R3D-18	77.3	47.5
RTT [29]	2020	K400	16×112^2	R3D-18	79.3	**49.8**
PCL (3DRotNet)		UCF	16×112^2	R3D-18	82.8	47.2
PCL (VCP)		UCF	16×112^2	R3D-18	83.4	48.8
PCL (VCP)		K400	16×112^2	R3D-18	**85.6**	48.0

network architectures, and different pre-trained datasets. The total duration of Kinetics-400 dataset is around 28 days while it is about one day for UCF101 datasets. Larger datasets, as well as input sizes, will usually boost the performance. In our experiments, we only set the input size of $16\times112\times112$ while we can achieve even better performance than methods such as SpeedNet [26] and MemDPC [37] when our PCL is pre-trained on UCF101 while they used larger pre-trained datasets, larger input size, and deeper networks. The best video recognition performance on the UCF101 dataset is achieved when our PCL is pre-trained on Kinetics-400, reaching 85.7%. On HMDB51, our best performance (48.8%) is obtained by using VCP as the

Table 4.7: Ablation studies on different kinds of combinations. Network architecture is based on R3D. Results are reported on UCF101 *split* 1. *Res* means using residual clip as input and *Aug* represents methods using strong data augmentations.

Exp.	Pretext	Contrastive	Res	Aug	Retrieval	Recog.
1	VCP	×	×	×	18.6	66.0
2	VCP	×	✓	×	25.6	77.0
3	×	✓	×	×	34.0	61.2
4	×	✓	✓	✓	44.7	79.3
5	VCP	✓	×	×	35.0	65.9
6	VCP	✓	×	✓	40.3	68.9
7	VCP	✓	✓	×	40.5	78.9
8	VCP	✓	✓	✓	**48.1**	**79.9**

pretext task baseline and ResNet-18-3D network backbone, outperforming all other methods except for RTT [29].

It may be claimed that in some papers, their proposed pretext task or contrastive learning methods were novel and could achieve state-of-the-art performance at that time. However, based on our experiments, we find there is much room for previous methods. Exploring the limits of each method and then conducting comparison may be a fair way.

4.5.3 Ablation Study: Effectiveness of Each Part

Because we have a lot of changes based on pretext tasks such as combining with contrastive learning, using residual clips, and data augmentation transformations, we want to find out how much impact each part contributes. We choose VCP as the pretext task baseline and R3D as the network backbone. Experiments are conducted on UCF101 *split* 1. Results are reported in Table 4.7. Because there are a lot of combination settings, we use the experiment ID to refer to for convenience. There are a total of 16 kinds of settings for all possible situations. Here, eight out of 16 are conducted because we think it is enough to show the effectiveness of each part in our proposal.

Residual clips. As we can see from the comparison pair, Exp. 1 and Exp. 2, by using residual clips instead of original RGB video clips, improvements can be obtained in both video retrieval and recognition. Similar performance can be found between Exp. 5 and Exp. 7, or Exp. 6 and Exp. 8.

Table 4.8: Ablation studies on the hyper-parameter α in Eq. 4.4. Network architecture is based on R3D and the pretext task is VCP. Results are reported on UCF101 *split* 1.

α	Top1	Top5	Top10	Top20	Top50	Recognition
0.1	43.2	63.1	72.7	80.9	89.8	**80.1**
0.5	**48.1**	64.7	**73.9**	82.0	**90.6**	79.9
1.0	**48.1**	**65.8**	73.6	**82.1**	90.0	79.3
10	45.9	65.0	72.7	81.1	89.6	73.5

Data augmentation. We can see from Exp. 7 and Exp. 8, with strong data augmentation transformations, the top-1 performance in video retrieval can be lifted from 40.5% to 48.1%. And 1% point improvement can be obtained in video recognition. From Exp. 5 and Exp. 6, we can also find that strong data augmentation is effective.

Methods combination. We can see a comparison set {Exp. 1, Exp. 3, Exp. 5}, whose experimental settings do not use our data processing strategies, a combination of VCP and contrastive learning can boost the performance of each. For comparison pair, Exp. 4 and Exp. 8, improvements can be also obtained when contrastive learning is combined with VCP in both video retrieval and recognition.

4.5.4 Ablation Study: Loss Weight Balancing

We conducted several experiments on loss weight balancing to explore the impact of α in Eq. 4.4. Experiments are conducted on the basis of pretext task VCP and the network backbone is R3D. Results are reported on UCF101 *split* 1 in both video retrieval and recognition.

We can see from Table 4.8, the retrieval performances are comparable when α is set to 0.5 or 1.0, higher than others. However, the best recognition result is achieved when α is set to 0.1. Compared with the setting $\alpha = 0.1$, the top-1 retrieval accuracy is 4.9% points higher for $\alpha = 0.5$ while its corresponding recognition accuracy is 0.2% points lower. To balance the performance in both video retrieval and recognition, we choose to set α to 0.5 for all of our experiments.

4.6 Discussions

In addition to the improvements on numbers, we would like to pose discussions on how a combination of pretext tasks and contrastive learning can yield better

performance. In this section, we show some evidence and analyses towards the combination and try to explore the potential mechanism behind it.

4.6.1 General Analysis

The mechanism of pretext tasks is not well explained in theory. Researchers aim to design tasks related to their final target tasks. For example, action retrieval and action recognition require temporal information to distinguish between samples. Thus, temporal-related tasks have been proposed. However, for individual pretext tasks, it is not clear which is the best, except based on particular performance metrics.

For contrastive learning, the basic idea is to distinguish one sample from another. However, determining why it functions well for motion representation extraction is difficult because spatial information may sometimes be enough. And same action clips in different instances will be treated as negatives during training.

Owing to many unclear issues, it is difficult to model the training target in a clear way. However, from the optimization target, we know that **pretext tasks focus within the sample while contrastive learning methods try to distinguish one instance from another**. By combining them together, the model can not only capture temporal information constrained by pretext tasks but also learn discriminative features from samples constrained by contrastive learning.

4.6.2 Feature Visualizations

To better understand learned features, we visualize them using t-SNE [142] in Fig. 4.5. Four different methods are used here: 1) random initialization, 2) one pretext task method, 3) one contrastive learning method, and 4) our proposed PCL. All models are trained in a self-supervised manner, except for the random initialization because it is initialized without training. The first ten categories in UCF101 *split* 1 are visualized and each point represents one video.

As we can see from Fig. 4.5, without any training, features are randomly distributed in the space. In the visualization of VCP and contrastive learning, features of the same class (in the same color) distribute more concentrated. With our PCL, it appears that the number of points is fewer because features of the same class are more close to each other and can be better clustered than the other three methods.

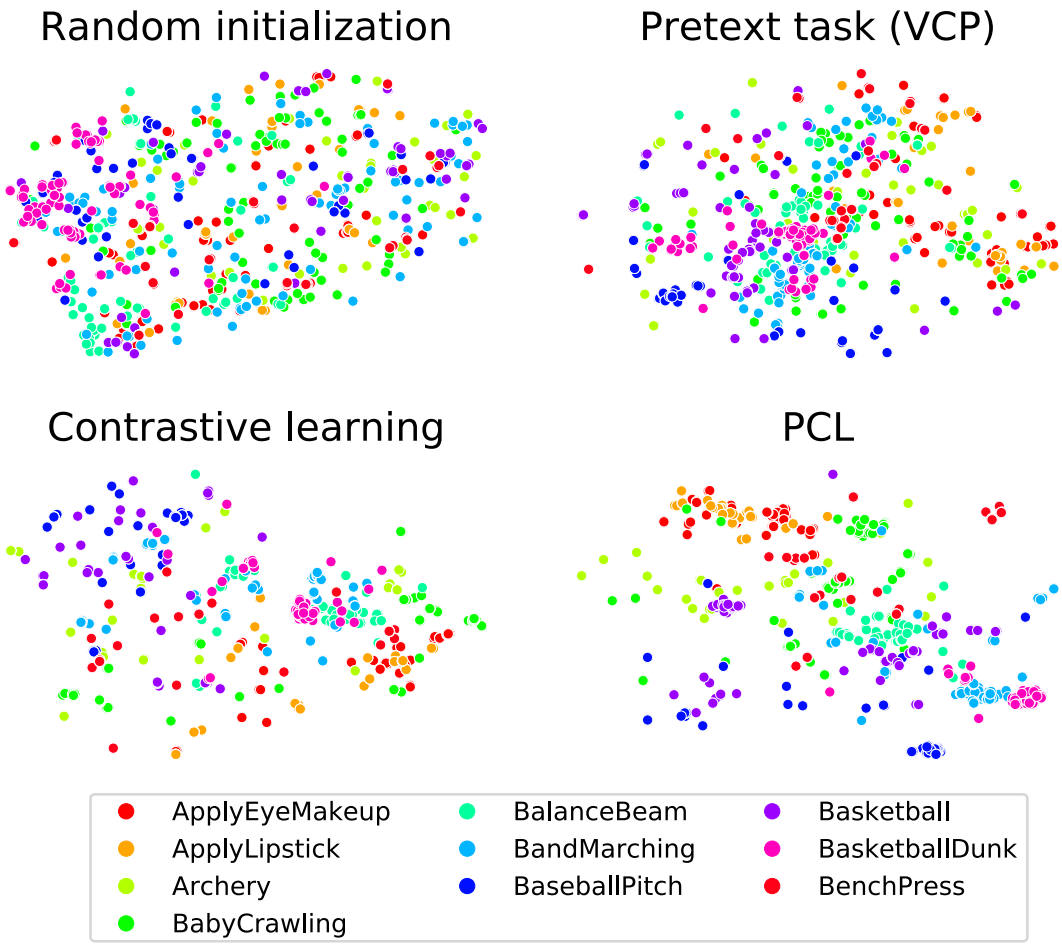


Figure 4.5: Visualizations using t-SNE. The point number for PCL appears smaller because points with the same color (i.e., the same action labels) are more concentrated. The first ten categories (in alphabetical order) in UCF101 are visualized.

4.6.3 Case Studies

To evaluate the advantage of pretext task, contrastive learning, and our PCL respectively, we use self-supervised trained models without changing parameters by fine-tuning. Therefore, video retrieval is used as the evaluation task. And all models are based on the R3D network backbone.

One combination of action categories in the UCF101 dataset is *Playing Musical Instruments*, where many similar actions are classified into different classes because of the different instruments. Therefore, contrastive learning should have better performance because it is constrained by distinguishing one sample from another, mainly based on spatial differences. Fig. 4.6 illustrates this trend that contrastive

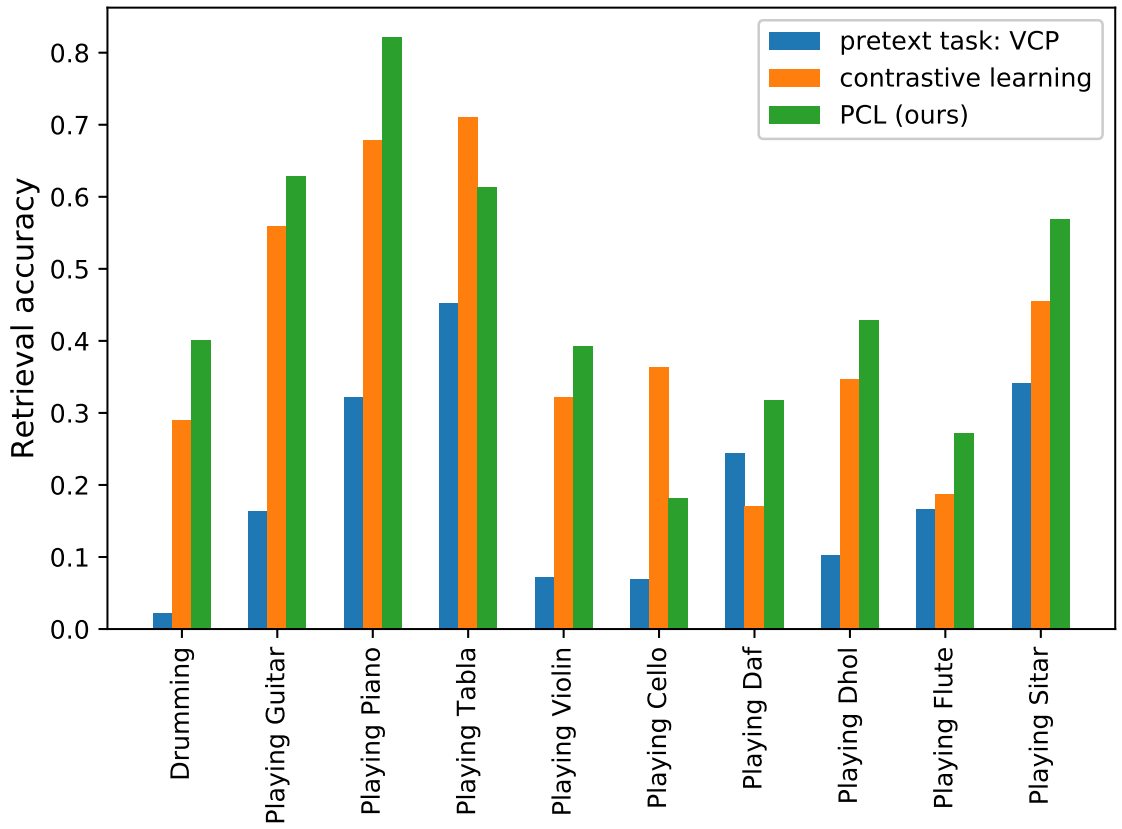


Figure 4.6: Video retrieval performance on each class. All classes here belong to the *Playing Musical Instruments* category. Our PCL can take the advantage of contrastive learning and compensate for pretext task baseline.

learning performs better than single pretext task, VCP. Though VCP utilized temporal transformations, the movements in many cases in this category are highly similar. Because our PCL is a combination of pretext tasks and contrastive learning, we can see that PCL can avoid the disadvantage of pretext tasks and even have better performance than the contrastive learning method. Note that in Fig. 4.6, for category *Playing Tabla* and *Playing Cello*, the contrastive learning method has better performance than our PCL. We find that for category *Playing Tabla*, the total number of testing cases is only 31, where 3 cases can cause around 10% points decrease. For category *Playing Cello*, 13.6% testing cases for our PCL are confused with category *Nunchuncks*, whose videos share similar composition with *Playing Tabla*. Though it is hard to say our PCL is the best for all cases, we can still say that our combination is generally better than pretext task or contrastive learning methods when they stand alone.

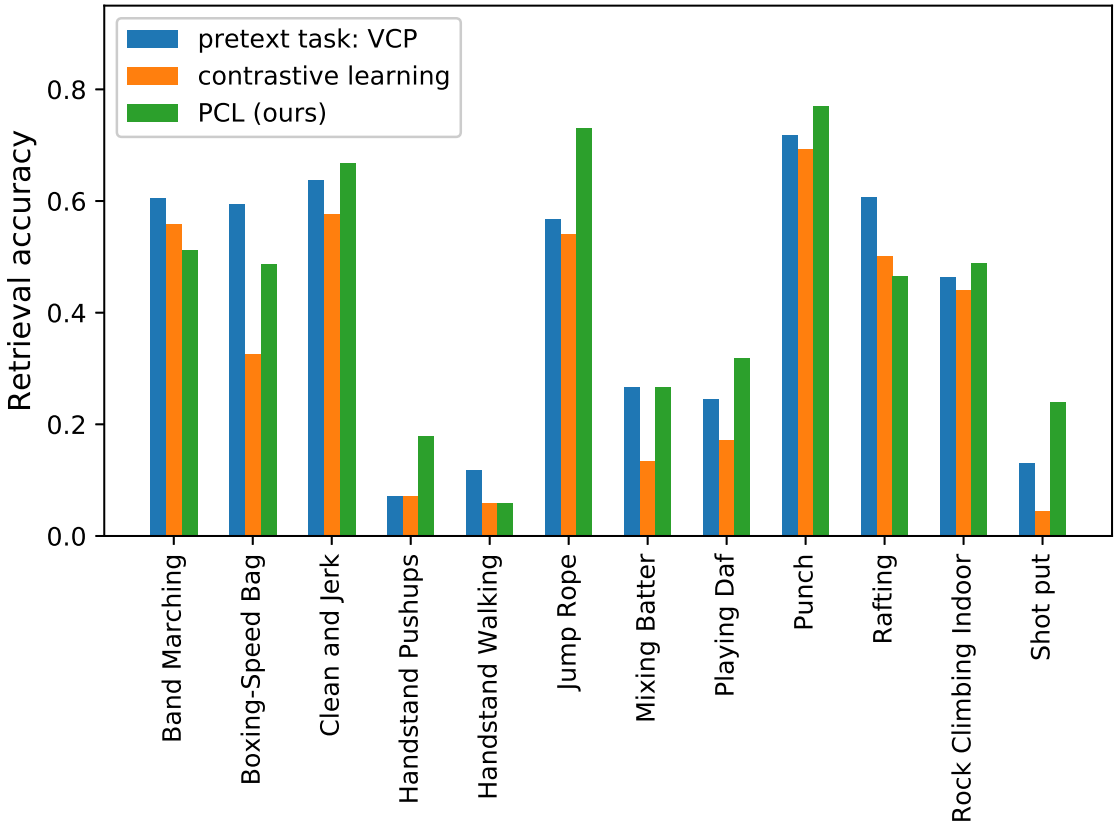


Figure 4.7: Video retrieval performance on each class. The classes here are those where pretext task method perform better than contrastive learning method.Our PCL can take the advantage of pretext task baseline and compensate for contrastive learning baseline.

Though from Table 4.7, we can see that the contrastive learning method can have much better performance than the pretext task VCP (44.7% vs 25.6%), there are still some advantages that pretext task has over contrastive learning. We visualize these classes where the pretext task method performs better than contrastive learning. As we can see from Fig. 4.7, some of these categories are much more temporal related. For example, the action *Clean and Jerk* represents a series of movements (Fig. 4.8). Because the pretext task VCP contains temporal transformations, it enables the model to capture more temporal-related features. And we can also see from the histogram that our PCL can achieve better or comparable performance.

In the tiny experiments, our PCL can achieve the best performance in 72 out of 101 classes, and the best averaging results, revealing that a combination of pretext task and contrastive learning can take the advantage of both.

Figure 4.8: Sample frames for action category *Clean and Jerk*, extracted from v_CleanAndJerk_g11_c03.avi in UCF101 dataset.

Table 4.9: Correlation coefficient based on pre-category video retrieval accuracies. Network backbone is R3D and models are all trained on UCF101 split 1 in the self-supervised way.

	Order	Rotation	Speed	Contrast
Order	1	0.5759	0.7678	0.5389
Rotation	-	1	0.6731	0.8307
Speed	-	-	1	0.6171
Contrast	-	-	-	1

4.6.4 Task Relation Exploration

For the baseline methods we used in our PCL, namely contrastive learning, video clip order prediction (VCOP [5]), video clip rotation detection (3DRotNet [4]), and video cloze procedure (VCP [3]), pretext tasks contain several transformations such as temporal order shuffling, rotation, permutation. As we introduced in Sec. **??**, another widely used pretext task is video playback rate recognition, though it has been combined with contrastive learning in [102]. Here, we want to explore the relations between these typical task solutions in video self-supervised learning. The connections between different tasks may help us understand the mechanism behind PCL.

We trained single models based on these typical self-supervised learning tasks in videos: order classification, rotation detection, speed recognition, and contrastive learning. To simplify, we denote them as *Order*, *Rotation*, *Speed*, and *Contrast* for short. For these trained models, we treated them as feature extractors and applied them to the video retrieval task. The correlation coefficient indexes are calculated across the per-category accuracies. In this way, model parameters are not changed, which can reflect the task relations to some extent.

Results are reported in Table 4.9. As we can see from the table, the correlation between Order and Speed is very high, indicating that with these two pretext tasks, model behaviors are similar. According to their definitions, they care more about

Table 4.10: Treat IIC as the contrastive learning method in PCL. Results are reported on UCF101 *split* 1 in video retrieval and recognition task.

Method	Backbone	Top1	Top5	Recognition
PCL (VCP + contrast)	R3D	48.1	64.7	79.9
IIC (rotate)	R3D	**53.1**	**70.1**	77.8
PCL (VCP + IIC)	R3D	48.2	65.8	**80.5**

temporal-related features. On the contrary, the correlation between Rotation and Contrast is high, implying they may help capture spatial-related features. A combination of different pretext tasks and contrastive learning can yield good performance because with additional constraints, the model has to learn more general video representations, ensuring higher performance when applied to downstream tasks. The VCP baseline is a combination of spatial and temporal transformations, making it the most effective among our pretext baselines. With additional supervision from contrastive learning, video representations should also become more discriminative. Thus, better video representations can be achieved via this kind of simple combination in our PCL.

4.6.5 Combination with Inter-Intra Contrastive Learning

In this chapter, we focus on video self-supervised learning, which is the same topic like that in Chapter 3 while we address different points. Both methods have made use of our solution in Chapter 2. Because PCL is a framework which combines contrastive learning with pretext tasks, and our IIC is one kind of contrastive learning, here we make use of them together. Results are reported in Table 4.10.

As we analyzed in Chapter 3, IIC is better than traditional contrastive learning for video self-supervised learning. Therefore, when replacing the contrastive learning method in PCL with IIC, 0.1% point and 0.6% point improvements can be obtained in video retrieval and action recognition tasks, respectively. Although the retrieval performance is not as good as using IIC only, much better performance can also be observed in action recognition (80.5% vs 77.8%). The conclusion can not be simply drawn about which is better between IIC and PCL (VCP + IIC). However, we can generally say that IIC is a better solution than traditional contrastive learning and in our PCL, stronger components will result in higher accuracies in downstream tasks in video self-supervised learning. Both are good solutions for video representation learning.

4.6.6 Limitations

One limitation of this work is the lack of solid explanations about the mechanism. Usually, a combination of multi-task learning can boost the performance, and different combinations of pretext tasks may also have effects. However, we would like to argue that one of our baselines is already a combination of several pretext tasks, and the performance will be further improved when combined with contrastive learning. Another limitation is the novelty because we do not propose any new pretext tasks. It is more like empirical studies in settings for video self-supervised learning. We hope this work paves the way for future work as stronger baseline settings in this direction. We have conducted experiments to try to find relations between typical pretext tasks. However, we think it is still far from clear and requires deeper exploration, which is one of the limitations for almost all existing works.

4.7 Conclusions

In this section, we proposed Pretext-Contrastive Learning (PCL), a joint optimization framework facilitating both pretext tasks and contrastive learning, which is beyond a simple combination. Data processing strategies such as residual clips and strong data augmentations are used in our framework. Extensive ablation studies showed the effectiveness of each component in our proposal. Experiments using different pretext task baselines with different network backbones in different evaluation tasks on two benchmark datasets revealed the effectiveness and the generality of our proposal. With our PCL framework and the empirical settings, pretext tasks and contrastive learning can boost each other, and old benchmarking baselines can be lifted to a new level, which could provide a guideline for the self-supervised video representation community. Our proposed PCL is sufficiently flexible enough and can be easily applied to almost any existing pretext task or contrastive method.

Chapter 5

Conclusions, Limitations, and Future Directions

5.1 Conclusions

To obtain video representations in an efficient and effective way, we had a deep exploration of the extraction of spatio-temporal information in this thesis. Supervised and self-supervised video are two promising learning paradigms, and on the basis of the fact that current models will ignore some important temporal information, we proposed our solutions for more general and robust video representation. Our proposed methods tackled three different aspects: 1) a novel input modality compatible with various 3D CNNs, 2) a temporal constraint in contrastive learning, and 3) a joint optimization framework in self-supervised video representation learning. Two video understanding tasks (i.e., video recognition and retrieval) are used to evaluate the quality of extracted features.

In Chapter 2, we presented residual frames which is a better replacement of traditional input using 3D CNNs for temporal feature extraction. Traditional 3D CNN-based methods used RGB video clips (i.e., stacked RGB frames) to train, where spatial information played an important role. Instead of traditional input data, we stacked frame differences as the new input modality. This simple change on the input data can force the model to focus more on temporal information, with less loss on the appearance part. Extensive experiments and pieces of evidence showed that our model can capture better temporal clues, without introducing additional computation on optical flow. We also demonstrate the generalization ability of our proposal by applying it to other tasks. For some videos which require more spatial

information, we used an additional appearance path to form the two-path solution. The usage of our proposed solution can be abroad and we also used it in the other two works of the thesis.

In Chapter 3, we presented IIC, an inter-intra contrastive learning framework in video self-supervised learning by introducing intra-negative samples. Our framework is based on the contrastive learning method, while enhancing its ability in video representation learning in two points: generate intra-negative samples from the anchor and set them as negative data; apply strategies for better temporal clue caption. We showed that our framework can help the model extract discriminative temporal information without any changes in the network architecture. The visualizations, as well as analyses, showed remarkable improvements over the traditional contrastive learning baseline in self-supervised video representation learning.

In Chapter 4, we presented pretext-contrastive learning (PCL), a joint optimization framework to extract video representations in a self-supervised manner, by a well-designed combination between pretext tasks and contrastive learning. This was inspired by analyzing the key concern behind pretext tasks and contrastive learning. We showed that a simple combination is very useful, even surpassing very recent works while the baseline methods we used are from a few years ago. We showed interesting findings of the behaviors for pretext tasks, contrastive learning methods, and our combination. The outstanding performance based on different pretext tasks in video recognition and retrieval tasks proves that this combination is generally effective. And a stronger component in our PCL can result in better performance such as using IIC instead of traditional contrastive learning in PCL. We hope this kind of finding as well as our analyses pave the way for further works in video self-supervised learning, benefiting from different supervision signals.

5.2 Limitations

In this thesis, we focus on supervised learning and self-supervised learning for video representation and propose one solution in supervised learning and two solutions in self-supervised learning. We have discussed the limitations one by one in each chapter. Here, we will discuss the limitations that exist throughout all of our works.

One limitation is the computation complexity. We use 3D ConvNets for all of our work, with good performances in two video understanding tasks. Though we have reduced the computation complexity by taking advantage of frame transformations and can extract good performances without optical flow, as we have discussed in

Chapter 2, for better performances, an additional appearance path might also be necessary for self-supervised learning solutions. Also, the computational cost is high for 3D ConvNets when compared with 2D ConvNet-based methods.

Another limitation is the mechanism behind them. We would like to argue that this might be one of the limitations for all existing works because it is far from clear for deep learning-based methods. We have shown pieces of evidence that can support our statement more or less, and we hope our analyses can help for a better and deeper understanding in this research area to some extent.

Video representation learning is a wide-range topic for video understanding. We tackle two main tasks (i.e., action recognition and video retrieval). There are a variety of video understanding tasks such as video segmentation, video temporal localization, and action detection. Considering the situation, we have not validated the effectiveness of our proposals in other video understanding tasks, or different learning schemes. Hopefully, we have found some works [143, 144] that have made use of our proposals and applied them to other video understanding tasks such as domain adaptation [143]. We hope our work can pave the way for future work in many other video understanding research topics.

5.3 Future Directions

In video representation learning, there are many interesting topics for future research. We focus on some trends, list advantages and disadvantages, and discuss some directions.

Video processing backbones Owing to the weakness of temporal modeling in current network backbones in videos, there are many recent developments in network architecture designs, such as 2D CNN variants and 3D CNN variants. Compared to hand-crafted network architectures, network architecture search (NAS) technologies [145–147, 64, 63] have developed very quickly for more compact but effective and efficient networks. However, NAS requires pre-defined module space for exploration and the performance seems not to be that appealing. Transformers [72, 148, 149] have brought high attention not only for its attention mechanism but also the transformer encoder and decoder blocks. After applying successful solutions from NLP to computer vision, vision transformer as well as its follow-ups show remarkable performance in image tasks [73, 150–152] as well as video representation [75, 76, 78, 79]. Directly applying transformers to videos request a large amount of computation,

large datasets, and long-time training are necessary to ensure the performance. Efficiently making use of transformers in videos is a promising direction.

Multi-modality learning Videos contain not only frames, but also sounds and even texts or other metadata. Video representation should not be limited to visual features only. Although some multi-modality models [153, 154, 42, 155] have been proposed with a combination of visual data and sounds/texts. The combination of these kinds of representations is simple, which has not been well studied. A joint learning framework which utilizes all possible information of videos should be explored, coping with all possibilities that some videos have rich information while others may not, representing variant video in one feature space. However, current approaches cannot handle different videos well when the variety is large, especially when transferring to a different domain (dataset or tasks).

Efficient Models for Deployment An enormous number of research papers have been published with better performance in different benchmark datasets. However, some methods are proposed without considering the model complexity and far from deployment for real products due to the high computational costs. To increase efficiency in the inference period, in addition to the network design, some technologies can be used such as quantization [156–158], knowledge distillation [159–161], as well as weight pruning [162–164]. There are a lot of works in natural language processing and computer vision tasks for images, while few address video approaches. Compared to images, this demand is more urgent for video representation for deployment because videos are naturally more complex than texts and images.

References

[1] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 618–626.

[2] Y. Tian, D. Krishnan, and P. Isola, "Contrastive multiview coding," *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 776–794, 2020.

[3] 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.

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

[5] 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/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10 334–10 343.

[6] 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 (ICCV)*, 2015, pp. 4489–4497.

[7] 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 (CVPR)*, 2018, pp. 6450–6459.

[8] K. Hara, H. Kataoka, and Y. Satoh, "Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 18–22.

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

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

[11] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "Slowfast networks for video recognition," pp. 6202–6211, 2019.

[12] C. Sun, A. Shrivastava, C. Vondrick, K. Murphy, R. Sukthankar, and C. Schmid, "Actor-centric relation network," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 318–334.

[13] J. Jiang, Y. Cao, L. Song, S. Z. Y. Li, Z. Xu, Q. Wu, C. Gan, C. Zhang, and G. Yu, "Human centric spatio-temporal action localization," in *ActivityNet Workshop on CVPR*, 2018.

[14] C. Gu, C. Sun, D. A. Ross, C. Vondrick, C. Pantofaru, Y. Li, S. Vijayanarasimhan, G. Toderici, S. Ricco, R. Sukthankar *et al.*, "Ava: A video dataset of spatio-temporally localized atomic visual actions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 6047–6056.

[15] Z. Li, Y. Huang, M. Cai, and Y. Sato, "Manipulation-skill assessment from videos with spatial attention network," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019, pp. 0–0.

[16] Y. Tang, Z. Ni, J. Zhou, D. Zhang, J. Lu, Y. Wu, and J. Zhou, "Uncertainty-aware score distribution learning for action quality assessment," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9839–9848.

[17] Y. Jiang, K. Cui, B. Peng, and C. Xu, "Comprehensive video understanding: Video summarization with content-based video recommender design," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019, pp. 0–0.

[18] J. Gao, X. Yang, Y. Zhang, and C. Xu, "Unsupervised video summarization via relation-aware assignment learning," *IEEE Transactions on Multimedia*, 2020.

[19] 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.

[20] K. Xu, X. Jiang, and T. Sun, "Two-stream dictionary learning architecture for action recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 3, pp. 567–576, 2017.

[21] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2016, pp. 1933–1941.

[22] N. Crasto, P. Weinzaepfel, K. Alahari, and C. Schmid, "Mars: Motion-augmented rgb stream for action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 7882–7891.

[23] 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.

[24] 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 (CVPR)*, 2017, pp. 3636–3645.

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

[26] 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.

[27] H. Cho, T. Kim, H. J. Chang, and W. Hwang, "Self-supervised spatio-temporal representation learning using variable playback speed prediction," *CoRR*, vol. abs/2003.02692, 2020.

[28] 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.

[29] S. Jenni, G. Meishvili, and P. Favaro, "Video representation learning by recognizing temporal transformations," *European Conference on Computer Vision*, 2020.

[30] J. Wang, J. Jiao, and Y.-H. Liu, "Self-supervised video representation learning by pace prediction," *European Conference on Computer Vision*, 2020.

[31] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, vol. 2, 2006, pp. 1735–1742.

[32] A. v. d. Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *CoRR*, vol. abs/1807.03748, 2018.

[33] R. D. Hjelm, A. Fedorov, S. Lavoie-Marchildon, K. Grewal, P. Bachman, A. Trischler, and Y. Bengio, "Learning deep representations by mutual information estimation and maximization," *International conference on learning representations*, 2019.

[34] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin, "Unsupervised feature learning via non-parametric instance discrimination," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3733–3742.

[35] 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.

[36] X. Chen, H. Fan, R. Girshick, and K. He, "Improved baselines with momentum contrastive learning," *CoRR*, vol. abs/2003.04297, 2020.

[37] T. Han, W. Xie, and A. Zisserman, "Memory-augmented dense predictive coding for video representation learning," *European Conference on Computer Vision*, 2020.

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

[39] 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," *Advances in Neural Information Processing Systems*, 2020.

[40] A. Owens and A. A. Efros, "Audio-visual scene analysis with self-supervised multisensory features," in *European Conference on Computer Vision*, 2018, pp. 631–648.

[41] B. Korbar, D. Tran, and L. Torresani, "Cooperative learning of audio and video models from self-supervised synchronization," in *Advances in Neural Information Processing Systems*, 2018, pp. 7763–7774.

[42] C. Sun, A. Myers, C. Vondrick, K. Murphy, and C. Schmid, "Videobert: A joint model for video and language representation learning," in *IEEE International Conference on Computer Vision*, 2019, pp. 7464–7473.

[43] T. Han, W. Xie, and A. Zisserman, "Video representation learning by dense predictive coding," in *IEEE/CVF International Conference on Computer Vision Workshop*, 2019, pp. 1483–1492.

[44] D. Tran, H. Wang, L. Torresani, and M. Feiszli, "Video classification with channel-separated convolutional networks," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 5552–5561.

[45] 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," *CoRR*, vol. abs/1705.06950, 2017.

[46] J. Carreira, E. Noland, C. Hillier, and A. Zisserman, "A short note on the kinetics-700 human action dataset," *CoRR*, vol. abs/1907.06987, 2019.

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

[48] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, "Temporal segment networks: Towards good practices for deep action recognition," in *European Conference on Computer Vision (ECCV)*, 2016, pp. 20–36.

[49] Y. Li, S. Song, Y. Li, and J. Liu, "Temporal bilinear networks for video action recognition," in *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, vol. 33, 2019, pp. 8674–8681.

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

[51] 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 (CVPR)*, 2018, pp. 7794–7803.

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

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

[54] 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.

[55] 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 (CVPR)*, 2018, pp. 6026–6035.

[56] K. Soomro, A. R. Zamir, and M. Shah, "Ucf101: A dataset of 101 human actions classes from videos in the wild," *CoRR*, vol. abs/1212.0402, 2012.

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

[58] 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 *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, vol. 1, no. 4, 2017, p. 5.

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

[60] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[61] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 1492–1500.

[62] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1–9.

[63] C. Feichtenhofer, "X3d: Expanding architectures for efficient video recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 203–213.

[64] W. Peng, X. Hong, and G. Zhao, "Video action recognition via neural architecture searching," in *2019 IEEE International Conference on Image Processing (ICIP)*, 2019, pp. 11–15.

[65] C. Yang, Y. Xu, J. Shi, B. Dai, and B. Zhou, "Temporal pyramid network for action recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 591–600.

[66] S. Zhang, S. Guo, W. Huang, M. R. Scott, and L. Wang, "V4d: 4d convolutional neural networks for video-level representation learning," 2020. [Online]. Available: https://openreview.net/forum?id=SJeLopEYDH

[67] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[68] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 2625–2634.

[69] 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.

[70] Y. Zhu, Z. Lan, S. Newsam, and A. Hauptmann, "Hidden two-stream convolutional networks for action recognition," in *Asian Conference on Computer Vision (ACCV)*, 2018, pp. 363–378.

[71] J. Stroud, D. Ross, C. Sun, J. Deng, and R. Sukthankar, "D3d: Distilled 3d networks for video action recognition," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2020, pp. 625–634.

[72] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[73] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *Advances in Neural Information Processing Systems*, 2020.

[74] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *International Conference on Machine Learning*. PMLR, 2021, pp. 10 347–10 357.

[75] Z. Liu, J. Ning, Y. Cao, Y. Wei, Z. Zhang, S. Lin, and H. Hu, "Video swin transformer," *CoRR*, vol. abs/2106.13230, 2021.

[76] A. Arnab, M. Dehghani, G. Heigold, C. Sun, M. Lučić, and C. Schmid, "Vivit: A video vision transformer," *CoRR*, vol. abs/2103.15691, 2021.

[77] H. Fan, B. Xiong, K. Mangalam, Y. Li, Z. Yan, J. Malik, and C. Feichtenhofer, "Multiscale vision transformers," *CoRR*, vol. abs/2104.11227, 2021.

[78] G. Bertasius, H. Wang, and L. Torresani, "Is space-time attention all you need for video understanding?" *International Conference on Machine Learning*, vol. 139, pp. 813–824, 2021.

[79] D. Neimark, O. Bar, M. Zohar, and D. Asselmann, "Video transformer network," *CoRR*, vol. abs/2102.00719, 2021.

[80] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2013, pp. 3551–3558.

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

[82] J. S. Pérez, E. Meinhardt-Llopis, and G. Facciolo, "Tv-l1 optical flow estimation," *Image Processing On Line*, vol. 2013, pp. 137–150, 2013.

[83] 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 (CVPR)*, 2009, pp. 248–255.

[84] K. Hara, Y. Ishikawa, and H. Kataoka, "Rethinking training data for mitigating representation biases in action recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, 2021, pp. 3349–3353.

[85] Z. Shi, C. Guan, L. Cao, Q. Li, J. Liang, Z. Gu, H. Zheng, and B. Zheng, "Cotere-net: Discovering collaborative ternary relations in videos," in *European Conference on Computer Vision (ECCV)*, 2020, pp. 379–396.

[86] A. Diba, M. Fayyaz, V. Sharma, M. Mahdi Arzani, R. Yousefzadeh, J. Gall, and L. Van Gool, "Spatio-temporal channel correlation networks for action classification," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 284–299.

[87] G. Yang and D. Ramanan, "Volumetric correspondence networks for optical flow," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019, pp. 794–805.

[88] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 2758–2766.

[89] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2462–2470.

[90] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "Models matter, so does training: An empirical study of cnns for optical flow estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 6, pp. 1408–1423, 2019.

[91] 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 (ACMMM)*, 2020, pp. 2193–2201.

[92] L. Tao, X. Wang, and T. Yamasaki, "Pretext-contrastive learning: Toward good practices in self-supervised video representation leaning," *arXiv preprint arXiv:2010.15464*, 2020.

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

[94] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2536–2544.

[95] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *European Conference on Computer Vision*, 2016, pp. 649–666.

[96] L. Tao, X. Wang, and T. Yamasaki, "Rethinking motion representation: Residual frames with 3d convnets," *IEEE Transactions on Image Processing*, vol. 30, pp. 9231–9244, 2021.

[97] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[98] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *International Conference on Learning Representations*, 2014.

[99] N. Komodakis and S. Gidaris, "Unsupervised representation learning by predicting image rotations," 2018.

[100] S. Guo, E. Rigall, L. Qi, X. Dong, H. Li, and J. Dong, "Graph-based cnns with self-supervised module for 3d hand pose estimation from monocular rgb," *IEEE Transactions on Circuits and Systems for Video Technology*, 2020.

[101] P. Xu, Z. Song, Q. Yin, Y.-Z. Song, and L. Wang, "Deep self-supervised representation learning for free-hand sketch," *IEEE Transactions on Circuits and Systems for Video Technology*, 2020.

[102] 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/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4006–4015.

[103] 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.

[104] 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," vol. 1, 2021.

[105] P. Sermanet, C. Lynch, Y. Chebotar, J. Hsu, E. Jang, S. Schaal, S. Levine, and G. Brain, "Time-contrastive networks: Self-supervised learning from video," in *2018 IEEE International Conference on Robotics and Automation*, 2018, pp. 1134–1141.

[106] A. Hermans, L. Beyer, and B. Leibe, "In defense of the triplet loss for person re-identification," *CoRR*, vol. abs/1703.07737, 2017.

[107] 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.

[108] X. Wang, H. Zhang, W. Huang, and M. R. Scott, "Cross-batch memory for embedding learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6388–6397.

[109] X. Wang, X. Han, W. Huang, D. Dong, and M. R. Scott, "Multi-similarity loss with general pair weighting for deep metric learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5022–5030.

[110] C. Feichtenhofer, H. Fan, B. Xiong, R. Girshick, and K. He, "A large-scale study on unsupervised spatiotemporal representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 3299–3309.

[111] H. Li, S. Yan, Z. Yu, and D. Tao, "Attribute-identity embedding and self-supervised learning for scalable person re-identification," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 10, pp. 3472–3485, 2019.

[112] H. T. Vu and C.-C. Huang, "Parking space status inference upon a deep cnn and multi-task contrastive network with spatial transform," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 4, pp. 1194–1208, 2018.

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

[114] 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.

[115] 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," pp. 11 804–11 813, 2021.

[116] X. Chen and K. He, "Exploring simple siamese representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15 750–15 758.

[117] Z. Tu, W. Xie, J. Dauwels, B. Li, and J. Yuan, "Semantic cues enhanced multimodality multistream cnn for action recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 5, pp. 1423–1437, 2018.

[118] J. Jiao, Y. Cai, M. Alsharid, L. Drukker, A. T. Papageorghiou, and J. A. Noble, "Self-supervised contrastive video-speech representation learning for ultrasound," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2020, pp. 534–543.

[119] S. Jeon, D. Min, S. Kim, and K. Sohn, "Mining better samples for contrastive learning of temporal correspondence," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1034–1044.

[120] C. Gan, T. Yao, K. Yang, Y. Yang, and T. Mei, "You lead, we exceed: Labor-free video concept learning by jointly exploiting web videos and images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 923–932.

[121] C. Gan, C. Sun, L. Duan, and B. Gon, "Webly-supervised video recognition by mutually voting for relevant web images and web video frames," in *European Conference on Computer Vision*. Springer, 2016, pp. 849–866.

[122] C. Gan, C. Sun, L. Duan, and B. Gong, "Webly-supervised video recognition by mutually voting for relevant web images and web video frames," in *European Conference on Computer Vision*, 2016, pp. 849–866.

[123] A. Iscen, G. Tolias, Y. Avrithis, and O. Chum, "Mining on manifolds: Metric learning without labels," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7642–7651.

[124] V. Verma, T. Luong, K. Kawaguchi, H. Pham, and Q. Le, "Towards domain-agnostic contrastive learning," in *International Conference on Machine Learning*. PMLR, 2021, pp. 10 530–10 541.

[125] M. Wu, C. Zhuang, M. Mosse, D. Yamins, and N. Goodman, "On mutual information in contrastive learning for visual representations," *CoRR*, vol. abs/2005.13149, 2020.

[126] M. Wu, M. Mosse, C. Zhuang, D. Yamins, and N. Goodman, "Conditional negative sampling for contrastive learning of visual representations," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=v8b3e5jN66j

[127] 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.

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

[129] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, pp. 91–99, 2015.

[130] 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.

[131] I. Loshchilov and F. Hutter, "Sgdr: Stochastic gradient descent with warm restarts," *International Conference on Learning Representations*, 2017.

[132] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9879–9889.

[133] 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*, 2020.

[134] M. Patrick, Y. M. Asano, R. Fong, J. F. Henriques, G. Zweig, and A. Vedaldi, "Multi-modal self-supervision from generalized data transformations," *CoRR*, vol. abs/2003.04298, 2020.

[135] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. Hinton, "Big self-supervised models are strong semi-supervised learners," *Advances in Neural Information Processing Systems*, 2020.

[136] 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*, pp. 1–1, 2021.

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

[138] F. Wang and H. Liu, "Understanding the behaviour of contrastive loss," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 2495–2504.

[139] 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.

[140] L. Jing and Y. Tian, "Self-supervised visual feature learning with deep neural networks: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 11, pp. 4037–4058, 2020.

[141] T. Wang and P. Isola, "Understanding contrastive representation learning through alignment and uniformity on the hypersphere," in *International Conference on Machine Learning*, 2020, pp. 9929–9939.

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

[143] X. Song, S. Zhao, J. Yang, H. Yue, P. Xu, R. Hu, and H. Chai, "Spatio-temporal contrastive domain adaptation for action recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 9787–9795.

[144] Y. Lin, X. Guo, and Y. Lu, "Self-supervised video representation learning with meta-contrastive network," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 8239–8249.

[145] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," *International Conference on Learning Representations*, 2017.

[146] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8697–8710.

[147] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le, "Regularized evolution for image classifier architecture search," in *Proceedings of the aaai conference on artificial intelligence*, vol. 33, no. 01, 2019, pp. 4780–4789.

[148] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 4171–4186, 2019.

[149] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner,

S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," vol. 33, pp. 1877–1901, 2020.

[150] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European Conference on Computer Vision*. Springer, 2020, pp. 213–229.

[151] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," *International Conference on Computer Vision (ICCV)*, 2021.

[152] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," *arXiv preprint arXiv:2111.06377*, 2021.

[153] M. Zolfaghari, Y. Zhu, P. Gehler, and T. Brox, "Crossclr: Cross-modal contrastive learning for multi-modal video representations," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 1450–1459.

[154] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," *International Conference on Machine Learning*, vol. 139, pp. 8748–8763, 2021.

[155] A. Miech, I. Laptev, and J. Sivic, "Learning a text-video embedding from incomplete and heterogeneous data," *CoRR*, vol. abs/1804.02516, 2018.

[156] M. Nagel, M. Fournarakis, R. A. Amjad, Y. Bondarenko, M. van Baalen, and T. Blankevoort, "A white paper on neural network quantization," *CoRR*, vol. abs/2106.08295, 2021.

[157] S. Shen, Z. Dong, J. Ye, L. Ma, Z. Yao, A. Gholami, M. W. Mahoney, and K. Keutzer, "Q-bert: Hessian based ultra low precision quantization of bert," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, 2020, pp. 8815–8821.

[158] O. Zafrir, G. Boudoukh, P. Izsak, and M. Wasserblat, "Q8bert: Quantized 8bit bert," *Workshop on Energy Efficient Machine Learning and Cognitive Computing - NeurIPS Edition*, pp. 36–39, 2019.

[159] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," *CoRR*, vol. abs/1910.01108, 2019.

[160] S. Sun, Y. Cheng, Z. Gan, and J. Liu, "Patient knowledge distillation for bert model compression," *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pp. 4322–4331, 2019.

[161] X. Jiao, Y. Yin, L. Shang, X. Jiang, X. Chen, L. Li, F. Wang, and Q. Liu, "Tinybert: Distilling bert for natural language understanding," *Findings of the Association for Computational Linguistics*, pp. 4163–4174, 2019.

[162] P. Michel, O. Levy, and G. Neubig, "Are sixteen heads really better than one?" *Advances in Neural Information Processing Systems*, pp. 14 014–14 024, 2019.

[163] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," *International Conference on Learning Representations*, 2019.

[164] A. Fan, E. Grave, and A. Joulin, "Reducing transformer depth on demand with structured dropout," *International Conference on Learning Representations*, 2019.

Publications

Publications related to the thesis

International Journal

[1] Li Tao, Xueting Wang, and Toshihiko Yamasaki, "Rethinking Motion Representation: Residual Frames with 3D ConvNets," in IEEE Transactions on Image Processing (TIP), vol.30, pp. 9231-9244, 2021. DOI: https://doi.org/10.1109/TIP.2021.3124156.

[2] Li Tao, Xueting Wang, and Toshihiko Yamasaki, "An Improved Inter-intra Contrastive Framework for Self-supervised Video Representation Learning," in IEEE Transactions on Circuits and Systems for Video Technology (T-CSVT), 2022. DOI: 10.1109/TCSVT.2022.3141051.

International Conference

[3] Li Tao, Xueting Wang, Toshihiko Yamasaki, Jingjing Chen, and Steven Hicks, "Reproducibility Companion Paper: Self-Supervised Video Representation Learning Using Inter-Intra Contrastive Framework," In Proceedings of the 29th ACM International Conference on Multimedia (ACMMM), pp. 3630–3632, 2021. DOI: https://doi.org/10.1145/3474085.3477939.

[4] Li Tao, Xueting Wang, and Toshihiko Yamasaki, "Self-Supervised Video Representation Learning Using Inter-Intra Contrastive Framework," In Proceedings of the 28th ACM International Conference on Multimedia (ACMMM), pp. 2193-2201, 2020. DOI: https://doi.org/10.1145/3394171.3413694.

[5] Li Tao, Xueting Wang, and Toshihiko Yamasaki, "Motion Representation Using Residual Frames with 3D CNN," In IEEE International Conference on Image Processing (ICIP), pp. 1786-1790, 2020. DOI: https://doi.org/10.1109/ICIP40778.2020.9191133.

Domestic Conference

[6] Li Tao, Xueting Wang, and Toshihiko Yamasaki, “An Improved Inter-Intra Contrastive Framework for Self-Supervised Video Representation Learning,” MIRU, L5-3, 2021.

[7] Li Tao, Xueting Wang, and Toshihiko Yamasaki, “Inter-Intra Contrastive Framework for Self-Supervised Spatio-Temporal Learning,” IEICE-PRMU, vol. 120, no. 300, PRMU2020-38, pp. 1-6, 2020.

[8] Li Tao, Xueting Wang, and Toshihiko Yamasaki, "Motion Representation Using Residual Frames with 3D ConvNets for Action Recognition," IEICE-IE, vol. 119, no. 422, IE2019-81, pp. 227-232, 2020.

[9] Li Tao, Xueting Wang, and Toshihiko Yamasaki, “Bag of Tricks for Video Recognition Using 3D Convolutional Neural Networks,” MIRU, PS1-32, 2019.

Publications non-related to the thesis

International Conference

[10] Yiyan Chen, Li Tao, Xueting Wang, and Toshihiko Yamasaki, “Weakly Supervised Video Summarization by Hierarchical Reinforcement Learning,” in Proceedings of the 1st ACM Multimedia Asia (ACMMM Asia), pp. 1-6, 2020.

Domestic Conference

[11] Xianliang Zhang, Li Tao, Xueting Wang, Toshihiko Yamasaki, “Better Temporal Representation for Unsupervised Video Summarization Based on Contrastive Self-Supervised Learning,” MIRU2021, L2-2, 2021.

[12] Shengzhou Yi, Li Tao, Xueting Wang, Toshihiko Yamasaki, “Class-Balanced Contrastive Pre-Training for Improving Long-Tailed Recognition,” MIRU, S5-5, 2021.

[13] Yiyan Chen, Li Tao, Xueting Wang, and Toshihiko Yamasaki, “Reinforcement Learning on Video Summarization with Hierarchical Structure,” IEICE-IE, vol. 119, no. 422, IE2019-89, pp. 305-310, 2020.

[14] Li Tao, Xueting Wang, Tatsuya Kawahara, and Toshihiko Yamasaki, “Television Advertisement Analysis Using Attention-based Multimodal Network,” JSAI, 1H4-OS-12b-01, 2020.

[15] Li Tao, Xueting Wang, Tatsuya Kawahara, and Toshihiko Yamasaki, "Improvement on Television Advertisement Analysis by Using Additional Text Information," IEICE-MVE, vol. 119, no. 190, MVE2019-15, 2019.

Awards

[16] Li Tao, MIRU2021学生奨励賞

Appendix A

Supplementary Materials on Residual Frames with 3D ConvNets

A.1 Code Sample to Generate Residual Frames

Here we show a simple usage to generate residual frames in our solution. Residual frames are stacked frame differences, and we can easily obtain frame differences by shifting video clips along temporal axis. In traditional 3D ConvNet-based methods, all we need is just to add one line code to transform RGB video clips to Residual ones. We show a code sample here. Our method can be easily embedded into any 3D ConvNet-based solutions in video understanding tasks to extract better video reprensentations.

```
# PyTorch style
"""
args:
    x     - one batch of videp clip data, in shape [B, C, T, H, W]
    model - the pre-defined 3D ConvNet
    y     - output of the model
"""
## One additional line to generate residual clips for 3D ConvNets
## Comment the following line for traditional RGB-input model
x = abs(x - torch.roll(x, 1, 2)) # Optional: x = x - torch.roll(x, 1, 2)

y = model(x)
```

Table A.1: Results on the UCF101 *split* 1. The network backbone is ResNet-18-3D.

Sampling rate	Modality	Pre-train	Clip acc	Top-1	Top-5
1	RGB	×	69.6	77.4	93.7
5	RGB	×	67.9	71.0	91.0
1	Res	×	72.3	79.5	94.2
5	Res	×	75.5	78.0	94.0
1	RGB	✓	81.2	88.4	97.9
5	RGB	✓	87.6	89.5	98.2
1	Res	✓	81.4	88.8	98.3
5	Res	✓	88.0	89.0	98.2

A.2 Ablation Study on Frame Sampling Rate

If the frames per second (FPS) shooting speed for videos is 30, then when decoding videos to raw video frames, one-second length video contain 30 frames. We call the sampling rate is 1 because one decoded frame is sampled from one raw frame. When we set the sampling rate to 5, it means that one decoded frame is sampled from every five frames. For our solution which use residual video clips in shape $16 \times 112 \times 112$, if the sampling rate is 1, this video clip covers around 0.5-second video. When the sampling rate is 5, one video clip will cover around 2.7-second video, containing more temporal movements. This is also an important factor because for some actions such as shooting, the movements are very fast while for some actions such as yoga, the movements are much slower.

In Sec. 2, we follow a standard data processing procedure as [8] to conduct all the experiments, while we further explore the effects made by the frame sampling rate here. Results are illustrated in Table A.1. Both traditional input modality and our proposed data modality are tested.

As we can see from the table, for different training settings (i.e., scratch training or fine-tuning, different data modalities), the performance differences exist. For example, for traditional RGB video clips, with dense sampling strategy (i.e., sample rate is 1), the top1 accuracy is 6.4% higher than that with sample rate 5 when models are trained from scratch. However, when models are fine-tuned from a pre-trained model weights, the performance of dense sampling strategy is 1.1% worse. A possible explanation is that appearance information is sufficient for a large amount

of cases because spatial information does not change too much in 0.5-second video. With knowledge from pre-trained model weights, additional temporal information can benefit. For our residual input model, the trend is similar but the gap becomes smaller.

An interesting finding is that when setting sampling rate to 5, the clip accuracy is much better for all cases except for RGB modality when the model is trained from scratch. Therefore, using clip representation might be a possible option to represent the whole video when models are obtained by fine-tuning. Because top-1 video action recognition accuracies are very similar for different sampling rate for our residual model, we choose to set sampling rate to 5 in all of our experiments.

Appendix B

Supplementary Materials on Inter-Intra Contrastive Learning Framework

B.1 Number of Negative Samples

We rewrite the contrastive learning loss function here,

$$\mathcal{L}_{contrast}^{v_i^1} = -\log \frac{h_\theta(\{v_i^1, v_i^2\})}{\sum_{j=1}^{k+1} h_\theta(\{v_i^1, v_j^2\}) + \sum_{j=1}^{k+1} h_\theta(\{v_i^1, v_j^{neg}\})}. \tag{B.1}$$

where k is the number of negative samples used in the calculation of contrastive learning. We have conducted experiments using different k. Results are illustrated in Table B.1.

Table B.1: Ablation studies on the number of k. R3D is used as the network backbone and frame repeating is used to generate intra-negative samples.

k	Top1	Top5	Recognition
512	51.9	67.7	76.6
1024	53.0	68.2	77.2
2048	52.0	67.6	76.6
4096	53.8	69.3	77.3

As we can see in Table B.1, the trend is not clear for the performance with different settings of k. In common sense in contrastive learning [38, 35], larger k indicating more negative samples in the constraint, and usually it will increase the variety in samples. However, when k is large enough, the differences become limited. In our case, we have extended negative samples with intra-negative video clips, and these intra-negative samples are high-quality negative samples compared to traditional negative samples. Thus, the improvements are limited when k is larger than 1024, even though the best setting for k is 4096.

B.2 Data Structure to Save Negative Samples

In Chapter 3, we use memory bank [34] technology to save negative features calculated in previous iterations during training. And when calculating contrastive loss, features are from the corresponding memory bank. The drawback is that the storage size for the memory bank will increase with larger and larger the dataset. SimCLR [38] did not make use of any additional memory. Instead, the negative samples are from the same batch because the batch size is very large. It is not acceptable for single GPU or even small clusters. In MoCo [35], a memory queue is used to save previous features, reaching a balance in memory occupation.

Our IIC is also compatible with all these settings, and we have also conducted experiments using a memory queue as MoCo. Results are in Table B.2.

Table B.2: Memory bank or memory queue. Network backbone is R3D and results are reported in UCF101 *split* 1 in video retrieval and recognition tasks.

Type	Intra-neg	Top1	Top5	Recognition
Memory bank	Repeat	53.0	68.2	77.2
Memory bank	Shuffle	49.4	65.4	76.5
Memory bank	Rotate	53.1	70.1	77.8
Memory queue	Repeat	43.0	60.9	75.4
Memory queue	Shuffle	49.1	67.0	77.6
Memory queue	Rotate	51.8	69.7	78.1

As we can see from the table, with memory bank, IIC can obtain better video retrieval performance for all three intra-negative generation options. For video recognition, it is interesting that the best performance is achieved by using a memory

Table B.3: Comparison with state-of-the-art methods in video retrieval on HMDB split 1. † indicates methods using optical flow in the training period. We highlight the best results in each block in **bold**.

Methods	Backbone	Top1	Top5	Top10	Top20	Top50
MemDPC [37]	R2D3D	7.7	25.7	40.6	57.7	-
MemDPC-Flow† [37]	R2D3D	15.6	37.6	52.0	65.3	-
CoCLR-RGB† [113]	S3D	**23.2**	**43.2**	**53.5**	**65.5**	-
VCOP [5]	C3D	7.4	22.6	34.4	48.5	70.1
VCP [3]	C3D	7.8	23.8	35.3	49.3	71.6
PRP [28]	C3D	10.5	27.2	40.4	56.2	75.9
PacePred [30]	C3D	12.5	32.2	45.4	61.0	80.7
IIC (repeat)	C3D	**20.0**	**43.0**	**56.6**	**70.5**	**86.1**
IIC (shuffle)	C3D	19.3	39.2	52.4	65.8	83.0
IIC (rotate)	C3D	19.5	44.7	58.7	73.1	89.3
VCOP [5]	R(2+1)D	5.7	19.5	30.7	45.6	67.0
VCP [3]	R(2+1)D	6.7	21.3	32.7	49.2	73.3
PRP [28]	R(2+1)D	8.2	25.3	36.2	51.0	73.0
PacePred [30]	R(2+1)D	12.9	31.6	43.2	58.0	77.1
IIC (repeat)	R(2+1)D	18.6	41.0	55.4	69.0	85.2
IIC (shuffle)	R(2+1)D	18.6	39.7	54.3	67.2	85.0
IIC (rotate)	R(2+1)D	**20.0**	**43.4**	**56.0**	**70.3**	**86.5**
3DRotNet	R3D-18	6.2	18.7	31.0	46.6	70.5
VCP [3]	R3D-18	10.9	25.2	36.8	51.5	71.8
PacePred [30]	R3D-18	9.6	26.9	41.1	56.1	76.5
IIC (repeat)	R3D-18	19.4	42.4	56.0	70.0	83.2
IIC (shuffle)	R3D-18	18.1	40.0	51.9	64.1	81.0
IIC (rotate)	R3D-18	**20.7**	**45.0**	**57.6**	**71.6**	**86.1**

queue. Taking different tasks into consideration, we can say that these two kinds of settings are comparable, revealing that IIC is compatible for MoCo training style.

B.3 Additional Retrieval Results on HMDB Dataset

We have shown retrieval results using R3D [8] as the network backbone. To validate the generalization ability for different network architectures, we show the performance using C3D [6], R(2+1)D [7], and R3D-18 [8] in Table B.3. CoCLR-RGB [113] can obtain the best performance in the table. However, it requires optical flow information during the training period, and the frame resolution is large than

ours (128 vs 112). It is clear that IIC is effective and can outperform other methods by a large margin.