Self-supervised visual learning by variable playback speeds prediction of a video

Cho, Hyeon; Kim, Taehoon; Chang, Hyung Jin; Hwang, Wonjun

DOI:
10.1109/ACCESS.2021.3084840

Document Version
Publisher's PDF, also known as Version of record

Citation for published version (Harvard):
Cho, H, Kim, T, Chang, HJ & Hwang, W 2021, 'Self-supervised visual learning by variable playback speeds prediction of a video', *IEEE Access*, vol. 9, 9443174, pp. 79562-79571. https://doi.org/10.1109/ACCESS.2021.3084840

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON DEEP LEARNING TECHNOLOGIES FOR INTERNET OF VIDEO THINGS

Received April 20, 2021, accepted May 19, 2021, date of publication May 28, 2021, date of current version June 8, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3084840

Self-Supervised Visual Learning by Variable Playback Speeds Prediction of a Video

HYEON CHO[1], TAEHOON KIM[1], HYUNG JIN CHANG[2], (Member, IEEE), AND WONJUN HWANG[1], (Member, IEEE)

[1]Department of Artificial Intelligence, Ajou University, Suwon-si 16499, South Korea
[2]School of Computer Science, University of Birmingham, Birmingham B15 2TT, U.K.

Corresponding author: Wonjun Hwang (wjwhang@ajou.ac.kr)

This work was supported in part by the Ministry of Science and ICT (MSIT), South Korea, through the Information Technology Research Center (ITRC) Support Program supervised by the Institute for Information & Communications Technology Planning & Evaluation (IITP), under Grant IITP-2021-2018-0-01431, in part by the National Research Foundation grant funded by the Korea Government (MSIT) under Grant NRF-2020R1F1A1066049, and in part by the BK21 FOUR Program of the National Research Foundation of Korea funded by the Ministry of Education under Grant NRF5199991014091.

ABSTRACT We propose *a self-supervised visual learning method* by predicting the variable playback speeds of a video. Without semantic labels, we learn the spatio-temporal visual representation of the video by leveraging the variations in the visual appearance according to different playback speeds under the assumption of temporal coherence. To learn the spatio-temporal visual variations in the entire video, we have not only predicted a single playback speed but also generated clips of various playback speeds and directions with randomized starting points. Hence the visual representation can be successfully learned from the meta information (playback speeds and directions) of the video. We also propose a new layer-dependable temporal group normalization method that can be applied to 3D convolutional networks to improve the representation learning performance where we divide the temporal features into several groups and normalize each one using the different corresponding parameters. We validate the effectiveness of our method by fine-tuning it to the action recognition and video retrieval tasks on UCF-101 and HMDB-51. [a]

[a]All the source code is released in https://github.com/hyeon-jo/PSPNet.

INDEX TERMS Action recognition, representation learning, self-supervised learning.

I. INTRODUCTION

The outstanding performance of image-based applications such as image recognition [1], object detection [2], and image segmentation [3] rely on large amounts of annotated data; for example, ImageNet [4], is used to train the deep-stacked layers of a convolutional neural network (CNN). However, when studying video recognition using deep learning, the availability of large sets of annotated data, such as Kinetics [5], is extremely costly and laborious. Therefore, an increasing need has arisen for a method that can be adapted to new domains without leveraging a huge amount of expensive supervision.

Recently, self-supervised learning has attracted increasing attention in many classification tasks such as image classification using a jigsaw puzzle [6], predicting the rotation of images [7], video classification based on prediction of the frame order [8] and clip order [9], and domain adaptation using self-supervised learning instead of the gradient reversal layer [10]. The self-supervised learning is a common learning framework that makes use of surrogate tasks that can be formulated without supervision by corresponding labels. The data itself could provide the supervisory signal for learning the image representation via self-supervised learning.

From the viewpoint of video classification, video data has its inherent characteristics such as spatial and temporal coherence, and most of the video-based self-supervised learning methods leverage spatio-temporal representation learning based on the steady chronological order of video data to understand the underlying temporal dynamics [8], [9], [11]. Specifically, the frames selected at equal temporal intervals are shuffled and their chronological order predicted [8],

The associate editor coordinating the review of this manuscript and approving it for publication was Jinjia Zhou.

whereas clips instead of video frames are used to leverage the inner steady temporal dynamics [9]. However, these previous methods have learned the visual representation of video via the temporal dynamics, but they make use limited data randomly selected from the entire video. Although this approach is advantageous in that it strengthens learning efficiency without supervision, the reliance on limited data is a constraint that has hampered further improvements in video classification performance.

In this paper, we propose a new RGB-based pretext task of self-supervised learning that predicts the order of various speeds and directions of video. In addition, we also suggest layer-dependable temporal group normalization (TGN) that helps 3D ConvNets learn better temporal dynamics by replacing vanilla batch normalization (BN). First, the video data are played repeatedly from the random start points at various predefined speeds including fast forward playback or backward playback. Each time the selected frames are collected to create a fixed-length clip. By recognizing the relative speeds of the clips, the variations in appearance and temporal coherence can be learned efficiently in the video over time without the semantic labels. Specifically, as shown in Fig 1, rather than simply predicting the chronological order of the frames, we formulate a task that predicts multiple playback speeds and then sorts the clips according to the playback speed. This approach enables the proposed method to learn the temporal video dynamics using as much frame data as possible from the video.

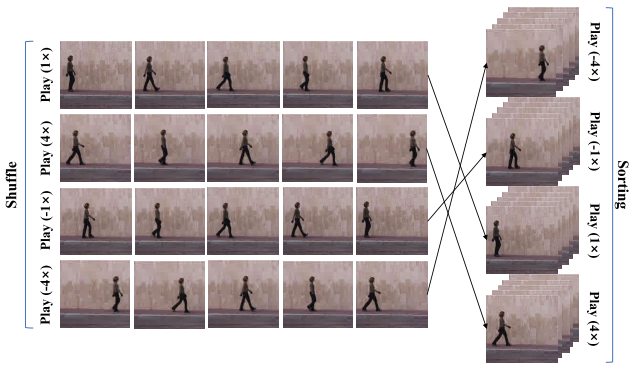


FIGURE 1. Conceptual illustration of the proposed method. We play back the video at variable speeds such as −4x, −1x, 1x and 4x with randomized starting points to create three clips. After shuffling the clips, we predict the playback speed of each clip and sort them according to the correct playback speeds. Once 3D ConvNets are able to solve this surrogate task without additional information, the network is ready to understand the temporal dynamics of the videos.

The main contributions of this paper can be summarized as follows:

- We introduce a self-supervised spatio-temporal representation learning via predicting the order of the multiple forward and backward playback speeds, which plays a pivotal role in generating many types of clips from videos and learning the spatio-temporal structure of videos without any manual annotations.
- We propose a new layer-dependable temporal group normalization method which enables efficient 3D ConvNet learning under the large variations of appearance and temporal coherence at videos.
- We show that using the proposed method improves the self-supervised learning performance for action recognition and video retrieval tasks through the extensive experiments on various 3D ConvNet architectures and datasets.

II. RELATED WORK

In this section, we review relevant literature on self-supervised learning and batch normalization. Self-supervised learning representation [6], [7], [12]–[15] has been studied for leveraging large-scaled training data without the label information in many vision applications such as image classification [6], Generative Adversarial Network [16] or Domain Adaptation [10] and video classification [9]. For this purpose, the surrogate tasks are formulated such that an inherent visual representation needs to be learned without a supervisory signal, and it plays a pivotal role in the accuracy of the vision applications.

A. SELF-SUPERVISED LEARNING FOR STATIC IMAGES

Self-supervised learning representation [6], [7], [12]–[15] has been studied for leveraging large-scaled training data without the label information in many vision applications such as image classification [6], Generative Adversarial Network [16] or Domain Adaptation [10] and video classification [9]. For this purpose, the surrogate tasks are formulated such that an inherent image representation needs to be learned without a supervisory signal, and it plays a pivotal role in the accuracy of the vision applications. Doersch *et al.* [12] designed a method to learn the representation of images by predicting the relative locations of two randomly sampled image patches. The rule of the jigsaw puzzle [6] has been generally employed for predicting a permutation of multiple randomly sampled patches. Another approach is identifying the randomly shuffled channels of the color image for image colorization [13]. The cue of the transitive in variance [14] could be used to match the patches of an image as for another self-supervised learning trial. Recently, Gidaris *et al.* [7] proposed a straightforward self-supervised learning method for predicting the angle of the random rotation transformation of an image and achieved good results in many ways.

B. SELF-SUPERVISED LEARNING FOR VIDEOS

Looking at the video-based self-supervised learning-based methods, temporal order verification [17] was early proposed for leveraging the temporal order of the sequential images because the video frames are stored in chronological order. It simply checked whether the temporal order is correct or not without the semantic labels. Soon after, Lee *et al.* [8] proposed the frame order sorting method, which increased the performance of the self-supervised learning method in video

recognition. They formulated the sequence sorting task as revealing the underlying chronological order of the sampled frames. In [18], the odd-one-out network has been proposed for self-supervised video representation learning, which identifies the odd temporal order of frames among some trials. Wang *et al.* [19] proposed the self-supervised learning method that learns visual features by regressing both motion and appearance statics along the spatial and temporal dimension. The jigsaw puzzle [6] was extended to the space-time cubic puzzles for training a 3D ConvNet in [20]. Xu *et al.* [9] have efficiently improved the frame order prediction method [8] by sorting the order of the neighboring clips as known as video clip order prediction (VCOP), where the clips are consistent with the video dynamics. In [11], the video cloze procedure (VCP) was proposed to learn the spatial-temporal representation of video data based on a method that uses spatial rotations and temporal shuffling method, which enhanced the accuracy in action recognition. Our proposed method is inspired by [8] and [9], but we make use of the playback speeds of the videos, not the correct sequential order of sampled frames [8] or clips [9]. This approach provides rich self-supervision to improve the learning performance based on the use of video. Inspired by SlowFast Network [21], the latest studies [22]–[24] on video speed have been conducted. [22] proposed a pretext method to predict the normal video speed to detect an unintentional event. In [23], they introduced a pretext for predicting the binary speediness (e.g., fast or normal speed) of moving objects in videos. Jenni *et al.* [24] showed a pretext learning not only the video speed but also temporal transformations. On the other hand, we design a pretext based on the fundamental temporal features of video playback speeds order prediction, e.g., playback speeds and directions.

C. NORMALIZATION METHODS

Batch normalization has shown considerable progress from the viewpoint of efficient learning of deep learning. After the success of batch normalization [25], [26], where the mean and variance are used for global normalization along the batch dimension, many normalization methods [27]–[30] have been proposed to improve the performance. First, weight normalization [29] is suggested normalizing the filter weights, and layer normalization [27] performed the normalization along the channel dimension only. Instance normalization [28] operates along with each instance sample. Group normalization [30] (GN) is a compromise between layer normalization and instance normalization where they proposed a layer that divides channels into groups and normalizes the features within each group. However, all these studies have limited performance in terms of the normalization along with channel, layer, or instance features, without deep consideration of the temporal features in the video. In this respect, we extend the group normalization to the layer-dependable TGN for video recognition with a novel task of self-supervised learning.

III. OUR APPROACH

We propose a surrogate task using variable playback speed prediction and 3D ConvNet using layer-dependable TGN to enable a large number of unlabeled videos to be used for efficient learning. When a 3D ConvNet is used to solve the Playback Speed Prediction (PSP) task, the 3D ConvNet successfully learns the fundamental visual representation of videos by understanding the temporal coherence changes according to the different playback speeds. For this purpose, as summarized in Fig 2, the proposed method consists of data sampling using playback speeds, the feature extraction with the proposed layer-dependable TGN, and the PSP network.

A. DATA SAMPLING USING VARIABLE PLAYBACK SPEEDS

From a single video, we sample frames based on the different playback speeds (e.g., from -5× to 5×) and generate clips of the various playback speeds with the sampled frames. Subsequently, the clips are randomly shuffled as inputs of the 3D ConvNet. Note that the starting frame is always set to an arbitrary position during sampling so that more diverse frames can be selected from one video. As described in [21], the multiple clips with different playback speeds could potentially allow the spatial semantics to be learned at slow speed and the motion dynamics to be learned at fast speed. Besides, because the clips are made by playing forward and backward at different speeds, it is possible to learn the temporal dynamics successfully even if the directional movements (e.g., push or pull) of the target in the video are fast or slow.

We define a tuple of n clips as L_n as follows:

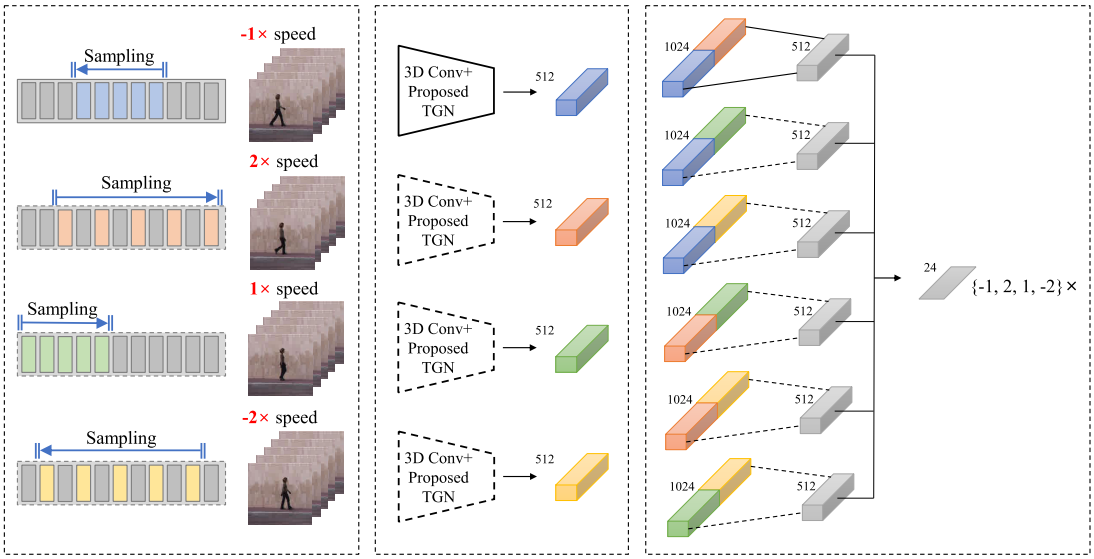
$$L_n = \begin{cases} \{l_{+3}, l_{-3}\} & \text{if } n = 2 \\ \bigcup_{i=1}^{(n-1)/2} \{l_{+1}, l_{+(2i+1)}, l_{-(2i+1)}\} & \text{if } n \text{ is odd} \\ L_{n-1} \cup \{l_{-1}\} & \text{if } n \text{ is even} \end{cases} \quad (1)$$

where n is the number of clips, and the subscript s of the clip l_s is the playback speed or the frame rate. Positive playback speed indicates fast forward playing, and negative speed is reverse playing or rewinding. When constructing a tuple of clips, we always ensure that half of the clips have positive speed and the other half have negative speed. Therefore, our formulation can accurately represent the direction of dynamics. In particular, our proposing method can distinguish between open and close dynamics which has been difficult to achieve so far [8]. In the order prediction, there are $n!$ (n factorial) possibilities in total. The clip, l, consists of m frames from random initial frame, f_i, (where f_i is the i^{th} frame of the original video), which is derived by

$$l_{+s} = \{f_i, f_{i+s}, f_{i+2s}, \ldots, , f_{i+(m-1)s}\}, \quad (2)$$

$$l_{-s} = \{f_{i-(m-1)s}, f_{i-(m-2)s}, \ldots, , f_i\}. \quad (3)$$

Unlike [9] we allow the same frames to be selected between the clips because of random initialization and the different frame rate associated with the clips. This allowance gives us a higher degree of freedom in using videos, so that more video clips can be used for learning a network model.



(a) Data Sampling using Playback Speeds (b) Feature Extraction with Proposed TGN (c) Playback Speed Prediction

FIGURE 2. The overall framework of the proposed method. (a) Data sampling using playback speeds. We sample frames according to the different playback speeds from the random initial points to form a clip and randomly shuffle them for 3D ConvNet. (b) Feature extraction with layer-dependable TGN. For efficient video learning, we train a 3D ConvNet using the proposed TGN instead of batch normalization. (c) Playback speed prediction. The extracted features are concatenated pairwise. The fully connected layers are used to encode the features, and the final layer uses these features to predict the playback speeds of the input clips. The dashed lines indicate that the network weights are shared with the straight lines.

B. FEATURE EXTRACTION USING LAYER-DEPENDABLE TGN

1) FEATURE EXTRACTION

The features of each sample clip are extracted by 3D ConvNet with shared parameters. We use three backbones as feature extractors, C3D [31], R3D, and R(2+1)D [32] to learn the spatio-temporal features effectively.

Our approach is to stack nine convolution layers in C3D and use the output of the Conv5b layer as a feature. The size of the 3D kernel is $3 \times 3 \times 3$. Both R3D and R(2+1)D consist of five blocks of convolution layers. The convolution block of R3D comprises two convolution layers with batch normalization and ReLU activation, respectively. The convolution block of R(2+1)D is similar to that of R3D except for the number of convolution layers. The last layer of the two models uses global adaptive pooling to extract spatio-temporal features.

2) LAYER-DEPENDABLE TGN

In video processing, the dependence between frames is not the same at all times. For example, the frame f_i at a specific time i has a higher correlation with $f_{(i+1)}$ than $f_{(i+k)}$ when $k \gg 1$. In other words, as time passes, the correlation of the preceding frame with the current frame is reduced and this characteristic is clearer when the video playback speed is fast. Note that our method uses fast playback as a task of self-supervised learning.

As shown in Fig 3 (a), the original batch normalization now obtains the mean and variance from each batch without considering frame changes over time thereby ignoring inherent spatio-temporal characteristics of the video. However, in this paper, we propose a layer-dependable temporal groups normalization method that normalizes individual groups divided along with temporal features, as shown in Fig 3 (b). Note that the number of temporal groups, p, depends on the depth of the corresponding layers, as shown in Table 1. Because we use the fixed temporal group feature size, g, and the temporal feature size, t, is expected to change according to the depth of the layers. In detail, The number of the temporal groups is decreased according to the increasing depth of the layers. From this scheme, we consider the many groups of features at the lower layers and the small groups of the feature at the higher layers when learning the 3D ConvNet. The features of the lower layers change with time while the features of the higher layer are processed for the target loss of action recognition. Its formulations for each channel indexed by α are as follows:

TABLE 1. C3D consists of five layers and the temporal group feature size is $g = 2$. After Conv1, Conv2, and Conv 3, each max pooling layer reduces the temporal feature size by half. t is a temporal feature size and p is a number of groups.

Layer	Conv1	Conv2	Conv3	Conv4	Conv5
t	16	8	4	2	2
p	8	4	2	1	1

$$\mu_i^{(\alpha)} = \frac{1}{bg_i hw} \sum_{j \in \{B, G_i, H, W\}} X_j, \tag{4}$$

$$\sigma_i^{(\alpha)} = \sqrt{\frac{1}{bg_i hw} \sum_{j \in \{B, G_i, H, W\}} (x_j - \mu_i)}, \tag{5}$$

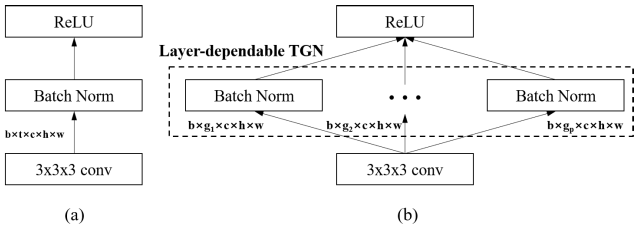


FIGURE 3. Network comparison. (a) Original batch normalization in video processing, and (b) proposed layer-dependable temporal groups normalization.

where $X \in \Re^{b\times g_i\times c\times h\times w}$ is the feature computed by a layer which is a 5D vector indexing the features in (B, G_i, C, H, W), and b, g_i, c, h, w denote the feature size of the mini-batch, i^{th} temporal group, channel, height, and width, respectively. Furthermore, $T = \{G_1, \ldots, G_p\}$ when T is the temporal feature, G_i is i^{th} temporal group feature, and p is the total number of temporal groups. Our proposed TGN computes μ and σ along the (B, G_i, H, W) axes for each channel. The general formulation of TGN feature normalization is as follows:

$$y_i^{(\alpha)} = \gamma_i^{(\alpha)}\hat{x}_i + \beta_j^{(\alpha)}, \tag{6}$$

where $\hat{X}_i^{(\alpha)} = \frac{(X_i - \mu_i^{(\alpha)})}{\sigma_i^{(\alpha)}}$. Specifically, pixels in the same group are normalized together by μ and σ. The TGN also learns the values of γ and β for each channel.

C. PLAYBACK SPEED ORDER PREDICTION

After extracting the features from the 3D ConvNet and the proposed TGN method, they are concatenated pairwise as shown in Fig 2 (c). We use a multi-layer perceptron to encode the pairwise concatenated features, which is a straightforward architecture for solving the order prediction problem [8], [9]. The final order prediction is then formulated using the softmax function with the concatenation of all pairwise features. We follow the protocol in [8] and in a single optimization step we always apply and predict all combinations for every clip in a mini-batch. Compared with [8], [9], the main difference is that we predict the order of multiple playback speeds instead of the order of frames or clips.

IV. EXPERIMENTAL RESULTS AND DISCUSSION

A. DATASETS

We evaluate our methods on two action recognition datasets, UCF-101 [33] and HMDB-51 [34] from the viewpoint of classification accuracy and retrieval performance. The UCF-101 dataset consists of 13,320 videos obtained from YouTube. These videos are classified into 101 classes. HMDB-51 is a dataset that expresses 51 human actions at least 101 times per class and consists of the total 6,849 clips. In this paper, we verify our method by using the clips for pre-training to compare the performance of the self-supervised learning methods [8], [9], [11]. Specifically, we train a 3D ConvNet using UCF-101 without label information first and fine-tune the model using labeled videos such as UCF-101 and HMDB-51, respectively. The network output is a 512-dimensional vector after the global spatio-temporal pooling layer, to which we append a fully-connected layer with softmax on top of it, as in [9] in the action recognition. The appended layer is only randomly initialized and the other layers are initialized from the self-supervised learning task.

1) IMPLEMENTATION DETAILS

All experiments are conducted using PyTorch [35]. We employ three well-known backbones, that is, C3D, R3D, and R(2+1)D. The input clips are resized and randomly cropped to 112 × 112. The clips are cropped in the center during the test procedure. When sampling the frames from a video, the starting position is set randomly and a total of $m = 16$ frames are extracted. To ensure that the 3D ConvNet is efficiently trained, which may be problematic in terms of memory consumption, we adopt SGD training on one GPU. The mini-batch size is $8\times n$, where n denotes the number of clips. The process of training the proposed PSP network continue for 300 epochs, following by another 150 epochs of fine-tuning to train the network to perform action recognition at a learning rate of 0.001. We use the momentum of 0.9, and weight decay of 0.0005. A dropout rate of 0.5 is used before the final fully connected layer.

B. ABLATION TEST

This section describes the validation of our method based on the C3D CNN using split 1 of UCF-101 for classification.

1) PLAYBACK SPEED PREDICTION NETWORK

We first investigate whether the prediction of the order of the variable playback speed clips differed from simple speed prediction. The simple speed prediction is performed using softmax, and a method of which the backbone architecture is the same as that of the proposed method is used. The total number of different speeds is three and the target speed ranging from $-3\times$ to $3\times$ is set randomly each iteration. Our method uses three different playback speeds such as $\{-3\times, 1\times, 3\times\}$. Table 2 compares the performance of two tasks. The accuracy of the proposed method is 5% higher as a result of learning more spatio-temporal information using several clips per iteration. Note that we cannot accurately predict the playback speed of the video by just watching the video; however, if the playback speeds of videos were to differ, we would easily be able to determine which video is faster or slower. In this respect, order prediction is a more appropriate task than speed prediction for self-supervised learning.

Next, we aim to explore the interrelation between playback direction and performance. The action recognition accuracy for the fast forward (67.25%), rewind (67.75%), and the combination of both (69.47%) of these playback methods in both directions are listed in Table 3. These results indicate that learning only single direction provide a similar accuracy, but the combined result is superior. Therefore, we can infer that the different playback directions provide a chance to

TABLE 2. Performance comparison between simple speed prediction and multiple playback speed order prediction. Both methods use three different speeds.

Task	Speed Pred.	Playback Speed Order Pred.
Accuracy	64.35	**69.47**

TABLE 3. Ablation study of playback directions. We use C3D with BN as the backbone. The model learns to predict the order of clips played at three different playback speeds. The clips were played back by using fast forward (FF) and rewind (RW) at three different speeds $s \in \{1, 3, 5\}$ in each playback direction.

Method	FF $\{1\times, 3\times, 5\times\}$	RW $\{-1\times, -3\times, -5\times\}$	FF+RW $\{-3\times, 1\times, 3\times\}$
Accuracy	67.25	67.75	**69.47**

TABLE 4. Comparison of action recognition accuracy by the playback speeds, s. In order to explore this ablation study, we set $n = 3$ and combination of two direction.

s	$\{-2\times, 1\times, 2\times\}$	$\{-3\times, 1\times, 3\times\}$	$\{-4\times, 1\times, 4\times\}$
Accuracy	67.50	**69.47**	68.81

TABLE 5. Performance changes with the number of clips. We use both forward and backward playback directions. VCOP is Video Clip Order Prediction [9]. For example, 2, 3, 4, 5, and 6 clips have $\{-3\times, 3\times\}$, $\{-3\times, 1\times, 3\times\}$, $\{-3\times, -1\times, 1\times, 3\times\}$, $\{-5\times, -3\times, 1\times, 3\times, 5\times\}$, and $\{-5\times, -3\times, -1\times, 1\times, 3\times, 5\times\}$, respectively.

# of clips	2 Clips	3 Clips	4 Clips	5 Clips	6 Clips
Ours	65.56	69.47	69.10	**71.70**	71.40
VCOP	67.80	67.93	66.77	61.67	-

learn different visual dynamics during training. For example, because walking backward involves the use of different muscles compared to walking forward [39], rewinding a video in which a person is walking forward would be different from playing a video in which someone is walking backward. In other words, rewinding the video could provide richer visual information by playing the fast forward. Consequently, our proposed method offers a way to learn more information at once by learning in both directions at the same time.

We test the effect of different playback speeds to determine the greatest relative speed difference for the best performance. As is clear from Table 4, an extremely small difference between playback speeds results in poor accuracy because the temporal dynamics are not significantly different between the $1\times$ and $2\times$ playback speeds. Moreover, learning from an excessively large difference in speed also does not benefit. Our method performs optimally when $s = 3$.

Finally, we evaluate different methods to understand the effect of the number of clips to determine the effect of variable n. The results in Table 5 indicate that the proposed method, the accuracy increased as the number of clips increased, but decreased again when six clips were used. This is largely because the number of possible orders in which the clips are arranged increased significantly; for example, for five and six clips have $5! = 120$ and $6! = 720$, respectively, in which case the complexity of the task is too large to train the model successfully. Compared with the VCOP method [9], The performance of the proposed method is superior ranging from 2 to 5 clips as a tuple set.

TABLE 6. Performance variation according to the number of elements in each groups. $m = 16$ and 3 clips are used.

Model	The element size of groups, g				
	16(=BN)	8	4	2	1
C3D	69.47	70.05	69.84	**70.26**	70.05
R3D	67.43	67.64	65.58	**69.10**	67.06
R(2+1)D	70.74	70.36	71.72	**72.92**	72.51

TABLE 7. Performance variation according to the number of clips. The effect of increasing the number of clips on the performance for $g = 2$ is determined.

Model	3 clips	4 clips	5 clips	6 clips
C3D	70.26	70.34	**71.53**	69.51
R3D	69.10	69.42	**69.44**	67.06
R(2+1)D	72.92	73.01	**74.65**	73.67

TABLE 8. Action recognition accuracy of variant 3D ConvNet based methods on HMDB-51 and UCF-101. The average accuracy is measured over three splits.

Dataset	HMDB-51			UCF-101		
Model	C3D	R3D	R(2+1)D	C3D	R3D	R(2+1)D
Random	24.7	23.4	22.0	61.8	54.5	55.8
VCOP	28.4	29.5	30.9	65.6	64.9	72.4
VCP	32.5	31.5	32.2	68.5	66.0	66.3
Ours	**34.31**	**33.68**	**36.82**	**70.44**	**68.98**	**74.82**

Specifically, the best accuracy, 67.93%, of VCOP at three clips is approximately 3.77% lower than the best performance overall, 71.70%, achieved by the proposed method for five clips. On the basis of this result, we conclude that the proposed method is more effective than the previous methods for different numbers of clips.

2) LAYER-DEPENDABLE TGN

We evaluate the performance improvement by the proposed TGN using backbones. We determine the optimal number of elements in the groups, and the results are presented in Table 6. All backbones achieve the best accuracy with $g = 2$ to normalize the output of each layer. Most of the results in Table 6 are higher (i.e., more accurate) than the BN baseline except for $g = 1$. Note that, at R(2+1)D, the proposed TGN improved the accuracy by 2.18% compared with the BN baseline. However, in case of C3D, performance improvement is marginal compared to R3D and R(2+1)D. C3D is difficult to train temporal representation because appearance and dynamics are jointly intertwined [32]. Therefore, TGN is more effective for R3D and R(2+1)D that can extract appearance and dynamics separately because TGN emphasizes temporal representation but cannot extract features.

Table 7 lists the effect of the number of clips on the action recognition performance, and the accuracy of the task using five clips is superior compared with that using three clips for backbones. Notably, the performance is saturated when the number of clips is over 5. The reason is that as the number of clips increases, the burden on the order prediction network increases. In conclusion, the proposed TGN if more

TABLE 9. Comparison with the published works on self-supervised learning using visual information. We indicate random cropping, scaling, random horizontal flip, and color jittering abbreviated as RC, S, HF, and CJ in the augmentation column, respectively. The number after the R3D is the number of the layers. The highest accuracy in each backbone architecture is indicated in bold.

Method	Backbone	Augmentation				Input size	Frames	Train DB	UCF-101	HMDB-51
		RC	S	HF	CJ					
SpeedNet [23]	S3D	✓			✓	224	64	Kinetics	81.1	48.8
PP [36]		✓	✓	✓	✓	224	64	UCF-101	87.1	**52.6**
CoCLR [37]		✓	✓	✓	✓	128	32	UCF-101	81.4	52.1
Ours		✓	✓	✓	✓	128	32	Kinetics	**88.7**	**54.9**
VCOP [9]	C3D	✓				112	16	UCF-101	65.6	28.4
VCP [11]		✓				112	16	UCF-101	68.5	32.5
Ours		✓				112	16	UCF-101	**70.4**	**34.3**
VCOP [9]	R3D-10	✓				112	16	UCF-101	64.9	29.5
VCP [11]		✓				112	16	UCF-101	66.0	31.5
Ours		✓				112	16	UCF-101	**69.0**	**33.7**
ST-Puzzle [20]	R3D-18	✓	✓	✓		224	16	Kinetics	65.8	33.7
DPC [38]		✓	✓	✓		128	25	UCF-101	60.6	-
		✓	✓	✓		128	25	Kinetics	68.2	34.5
TT [24]		✓	✓	✓	✓	128	16	Kinetics	79.3	**49.8**
Ours		✓	✓	✓	✓	128	16	Kinetics	**82.8**	48.8
VCOP [9]	R(2+1)D	✓				112	16	UCF-101	72.4	30.9
VCP [11]		✓				112	16	UCF-101	66.3	32.2
Ours		✓				112	16	UCF-101	**74.8**	**36.8**
PP [37]		✓	✓	✓	✓	112	25	Kinetics	77.1	36.6
TT [24]		✓	✓	✓	✓	128	16	UCF-101	81.6	46.4
Ours		✓	✓	✓	✓	128	16	Kinetics	**83.0**	**50.2**
Supervised	R3D-18	✓	✓	✓		112	16	Kinetics	83.5	53.6

effective for larger variations in the dynamics and appearance of the sampled clips, which is the consequence of the task responsible for the proposed PSP network.

C. ACTION RECOGNITION

In this section, we shows the effectiveness of our method in action recognition. We conduct experiments on UCF-101 and HMDB-51 and results are measured over three split of each dataset. In Table 8, we compare the proposed method with the latest self-supervised learning methods such as VCOP [9] and VCP [11] for both UCF-101 and HMDB-51. To validate the generality of the proposed method, we evaluate three backbone networks: C3D, R3D, and R(2+1)D in these experiments. Compared with random initialization (e.g., training from scratch) for 3D ConvNets, all self-supervision methods including ours exhibit superior performance. However, our method always outperforms the two other methods such as VCOP and VCP regardless of the 3D ConvNet models.

From table 9, our method shows a comparable results compared to the state-of-the-art (SOTA) self-supervised methods such as VCOP, VCP, Dense Predictive Coding (DPC) [38], SpeedNet [23], Temporal Transformation (TT) [24], Pace Prediction (PP) [36], and CoCLR [37]. Since each method has different backbone architecture, hyper-parameter setting, and augmentation method, we compare the performance of each backbone and augmentation setting to ensure fair comparison. Our method achieves the best results on almost backbones and augmentation settings over two datasets. The use of random crop only as an augmentation method is difficult to show good performance because of the high possibility of training shortcut, but our method shows about 2% to 5% improvement in all backbone compared to VCOP or VCP. In addition, our method shows higher accuracy than other methods using the same augmentation even when training shortcut is removed using color jittering or horizontal flip. To be noticed, our performance of R3D-18 as backbone is close to the supervised model using the pretrained on Kinetics. In addition, we make the best accuracy using S3D on UCF-101 and HMDB-51.

D. VIDEO CLIP RETRIEVAL

The performance of our method is confirmed by searching for nearest-neighbor video retrieval. The overall process of video retrieval used in the experiment followed that of [9], [40] and was evaluated with the split 1 of UCF-101 and HMDB-51 as in the previous papers. The first step of the entire experimental process entailed loading the weight of the trained model by using the training protocol presented in this paper. At this time, the feature was extracted by using 3D max pooling instead of the pre-existing global spatio-temporal pooling after passing through the final convolutional layer of the weight-loaded model. To measure the retrieval performance, we calculate the cosine distance between the testing and training video sets. The shortest nearest-neighbor video clip of k is found among the calculated cosine distances. If the correct answer is included among the k nearest neighbor video clips, the result is considered to be successful. After the video retrieval is performed for all test video sets, the accuracy is calculated by obtaining the number of correct answers divided by the total. In Fig 4, selected results of the video

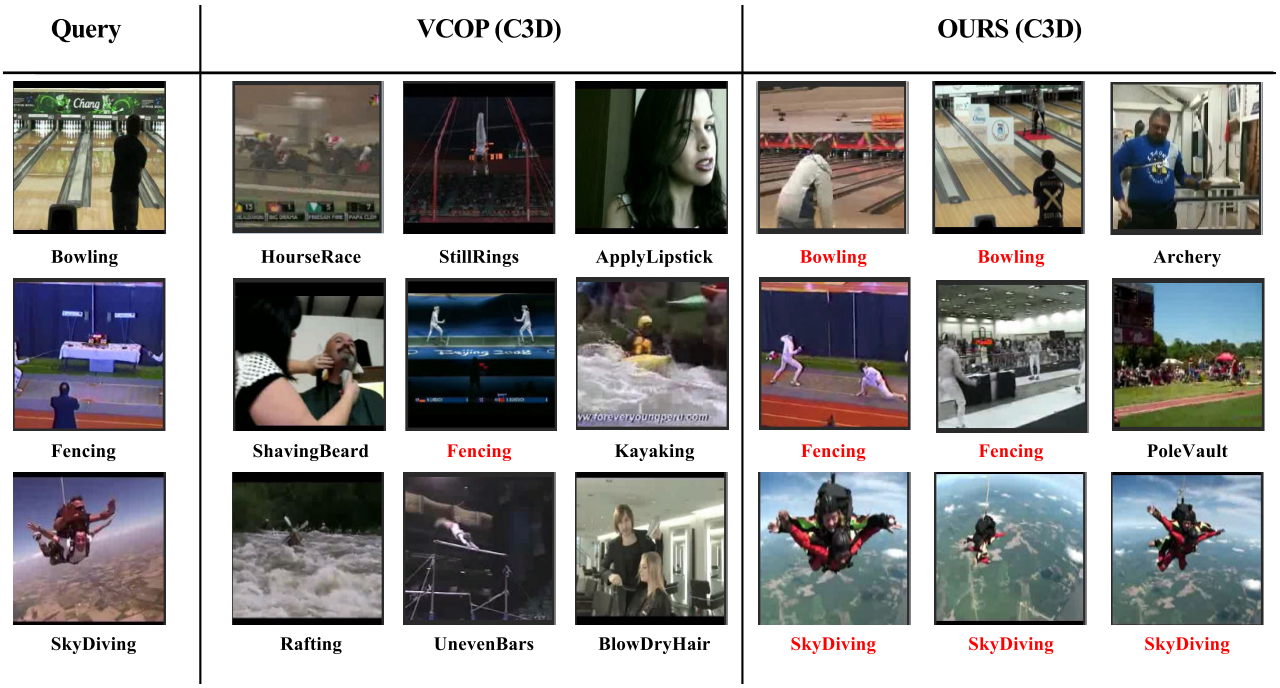


FIGURE 4. **Samples of video clip retrieval results. The labels highlighted in red indicate that this video clip is in the same category as the test video clip.**

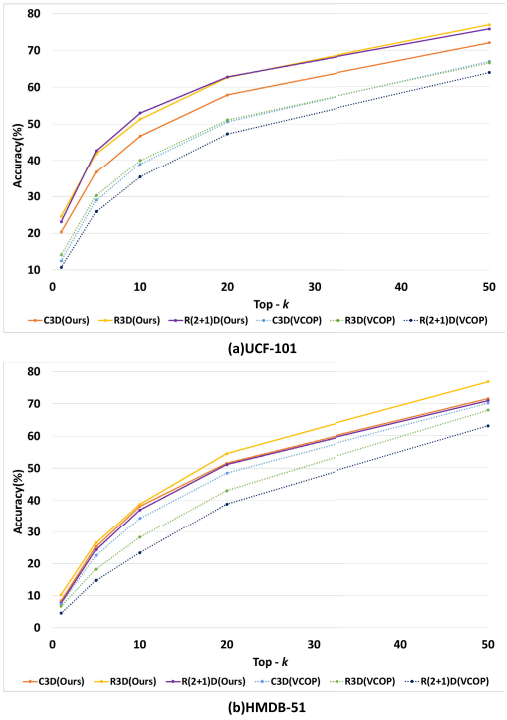


FIGURE 5. **Results of video retrieval. Our performances are compared with those of VCOP for C3D, R3D, and R(2+1)D networks.**

clip retrieval through samples are shown. Compared with the VCOP method, the qualitative results obtained with our method for the direction dynamics of videos such as bowling, fencing, and skydiving are more accurate.

The quantitative results are shown in Table 10. The proposed method achieves the highest accuracy among the

TABLE 10. **Video clip retrieval results on UCF-101 and HMDB-51. The backbone architectures of VCOP, VCP, and ours are 3D ConvNet based on the self-supervised learning method and we chose the best accurate network among C3D, R3D, and R(2+1)D.**

DB	Method	Top-K				
		1	5	10	20	50
UCF-101	VCOP	14.1	30.3	40.4	51.1	66.5
	VCP	19.9	33.7	42.0	50.5	64.4
	SpeedNet	13.0	28.1	37.5	49.5	65.0
	Ours	**24.6**	**41.9**	**51.3**	**62.7**	**67.9**
HMDB-51	VCOP	7.6	22.9	34.4	48.8	68.9
	VCP	7.8	23.8	35.3	49.3	71.6
	Ours	**10.3**	**26.6**	**38.8**	**54.6**	**76.8**

well-known self-supervised methods [6], [8], [9], [11], [40] through most ranks ranging from 1 to 50 at both UCF-101 and HMDB-51 databases. Specifically, the proposed method shows up to 5.7% and 2.8% higher accuracy than the state-of-the-art methods at the top-5 rank of UCF-101 and HMDB-51 datasets, respectively. Regardless of the types of the 3D ConvNets, this superiority is also confirmed by the ROC curves in Fig 5. We compared the results with VCOP for various 3D ConvNet architectures, the solid line is the PSP results that we propose, and the dotted line is the VCOP results. (a) shows a visualize for the top-k retrieval result in UCF-101 database and (b) shows a visualize for the top-k retrieval result in HMDB-51 database. From this result, we conclude that the proposed method works better than the well-known methods in the video retrieval task.

V. CONCLUSION

In this paper, We propose a various playback speed prediction network as a salient surrogate of the self-supervised learning

and a layer-dependable temporal group normalization for 3D ConvNet. Our method is able to train the natural visual temporal flow of videos as a pre-trained model by ordering the different fast forward playback speeds as well as the rewind speeds and designed the novel temporal group normalization for efficient visual representation learning at action recognition and video retrieval task. In the end, we have verified the superiority of our method by the extensive experiments.

REFERENCES

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[2] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.
[3] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 12, pp. 2481–2495, Dec. 2017.
[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.
[5] J. Carreira and A. Zisserman, "Quo vadis, action recognition? A new model and the kinetics dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6299–6308.
[6] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 69–84.
[7] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," 2018, *arXiv:1803.07728*. [Online]. Available: http://arxiv.org/abs/1803.07728
[8] H.-Y. Lee, J.-B. Huang, M. Singh, and M.-H. Yang, "Unsupervised representation learning by sorting sequences," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 667–676.
[9] D. Xu, J. Xiao, Z. Zhao, J. Shao, D. Xie, and Y. Zhuang, "Self-supervised spatiotemporal learning via video clip order prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 10334–10343.
[10] Y. Sun, E. Tzeng, T. Darrell, and A. A. Efros, "Unsupervised domain adaptation through self-supervision," 2019, *arXiv:1909.11825*. [Online]. Available: http://arxiv.org/abs/1909.11825
[11] 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 *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 11701–11708.
[12] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1422–1430.
[13] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 649–666.
[14] X. Wang, K. He, and A. Gupta, "Transitive invariance for self-supervised visual representation learning," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 1329–1338.
[15] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2536–2544.
[16] T. Chen, X. Zhai, M. Ritter, M. Lucic, and N. Houlsby, "Self-supervised GANs via auxiliary rotation loss," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 12154–12163.
[17] I. Misra, C. L. Zitnick, and M. Hebert, "Shuffle and learn: Unsupervised learning using temporal order verification," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2016, pp. 527–544.
[18] B. Fernando, H. Bilen, E. Gavves, and S. Gould, "Self-supervised video representation learning with odd-one-out networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3636–3645.
[19] 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 *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4006–4015.
[20] D. Kim, D. Cho, and I. S. Kweon, "Self-supervised video representation learning with space-time cubic puzzles," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 8545–8552.
[21] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "SlowFast networks for video recognition," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6202–6211.
[22] D. Epstein, B. Chen, and C. Vondrick, "Oops! Predicting unintentional action in video," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 919–929.
[23] 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 *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 9922–9931.
[24] S. Jenni, G. Meishvili, and P. Favaro, "Video representation learning by recognizing temporal transformations," in *Proc. Eur. Conf. Comput. Vis.*, Aug. 2020.
[25] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*. [Online]. Available: http://arxiv.org/abs/1502.03167
[26] S.-A. Rebuffi, H. Bilen, and A. Vedaldi, "Learning multiple visual domains with residual adapters," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 506–516.
[27] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," 2016, *arXiv:1607.06450*. [Online]. Available: http://arxiv.org/abs/1607.06450
[28] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Instance normalization: The missing ingredient for fast stylization," 2016, *arXiv:1607.08022*. [Online]. Available: http://arxiv.org/abs/1607.08022
[29] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 901–909.
[30] Y. Wu and K. He, "Group normalization," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 3–19.
[31] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4489–4497.
[32] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, "A closer look at spatiotemporal convolutions for action recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6450–6459.
[33] K. Soomro, A. R. Zamir, and M. Shah, "UCF101: A dataset of 101 human actions classes from videos in the wild," 2012, *arXiv:1212.0402*. [Online]. Available: http://arxiv.org/abs/1212.0402
[34] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "HMDB: A large video database for human motion recognition," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 2556–2563.
[35] P. Adam, G. Sam, C. Soumith, C. Gregory, Y. Edward, D. Zachary, L. Zeming, D. Alban, A. Luca, and L. Adam, "Automatic differentiation in PyTorch," in *Proc. Neural Inf. Process. Syst.*, 2017.
[36] J. Wang, J. Jiao, and Y.-H. Liu, "Self-supervised video representation learning by pace prediction," 2020, *arXiv:2008.05861*. [Online]. Available: http://arxiv.org/abs/2008.05861
[37] T. Han, W. Xie, and A. Zisserman, "Self-supervised co-training for video representation learning," 2020, *arXiv:2010.09709*. [Online]. Available: http://arxiv.org/abs/2010.09709
[38] T. Han, W. Xie, and A. Zisserman, "Video representation learning by dense predictive coding," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop (ICCVW)*, Oct. 2019.
[39] M. Lee, J. Kim, J. Son, and Y. Kim, "Kinematic and kinetic analysis during forward and backward walking," *Gait Posture*, vol. 38, no. 4, pp. 674–678, Sep. 2013.
[40] U. Buchler, B. Brattoli, and B. Ommer, "Improving spatiotemporal self-supervision by deep reinforcement learning," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 770–786.

HYEON CHO received the B.S. degree from the Department of Software and Computer Engineering, Ajou University, South Korea, in 2018, where he is currently pursuing the Ph.D. degree. He is also studying on improving the performance of the action recognition model. His current research interests include computer vision, pattern recognition, and deep learning.

TAEHOON KIM received the B.S. degree from the Department of Software and Computer Engineering, Ajou University, South Korea, in 2020, where he is currently pursuing the master's degree. He is also studying on video recognition using self-supervision manners. His current research interests include computer vision, pattern recognition, and deep learning.

HYUNG JIN CHANG (Member, IEEE) received the B.S. and Ph.D. degrees from the School of Electrical Engineering and Computer Science, Seoul National University, Seoul, South Korea. He was a Postdoctoral Researcher with the Imperial Computer Vision and Learning Laboratory and the Personal Robotics Laboratory, Department of Electrical and Electronic Engineering, Imperial College London. He is currently a Lecturer (equivalent to an Assistant Professor) with the School of Computer Science, University of Birmingham. His current research interests include human understanding through visual data analysis including human/hand pose estimation, eye gaze tracking, articulated structure learning, human–robot interaction, 6D object pose tracking, human action understanding, and user modeling.

WONJUN HWANG (Member, IEEE) received the B.S. and M.S. degrees from the Department of Electronics Engineering, Korea University, South Korea, in 1999 and 2001, respectively, and the Ph.D. degree from the School of Electrical Engineering, Korea Advanced Institute of Science and Technology (KAIST), South Korea, in 2016. From 2001 to 2008, he was a Research Staff Member with the Samsung Advanced Institute of Technology (SAIT), South Korea. He contributed to the promotion of Advanced Face Descriptor, Samsung and NEC joint proposal, to MPEG-7 international standardization, in 2004. He proposed the SAIT face recognition method which achieved the best accuracy under the uncontrolled illumination situation at Face Recognition Grand Challenge (FRGC) and Face Recognition Vendor Test (FRVT), in 2006. He developed the real-time face recognition engine for the Samsung cellular phone, SGH-V920, in 2006. From 2009 to 2011, he was a Senior Engineer with Samsung Electronics, South Korea, where he worked on developing face and gesture recognition methods for Samsung humanoid robot, a.k.a RoboRay. He rejoined the SAIT, as a Research Staff Member, in 2011. From 2011 to 2014, he worked for a 3D medical image processing with Samsung surgical robot. From 2014 to 2016, he worked on developing deep learning-based face detection and recognition methods with Samsung Galaxy series. He joined the Department of Software and Computer Engineering, Ajou University, South Korea, in 2016, where he is currently an Associate Professor. His research interests include computer vision, pattern recognition, and deep learning.