Pattern Recognition 156 (2024) 110804

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Repeat and learn: Self-supervised visual representations learning by Repeated Scene Localization

Hussein Altabrawee [a,b], Mohd Halim Mohd Noor [a,*]

[a] Universiti Sains Malaysia, School of Computer Sciences, Main Campus, Gelugor, 11800, Penang, Malaysia
[b] Al-Muthanna University, Computer Center, Main Campus, Samawah, 66001, Al-Muthanna, Iraq

ARTICLE INFO

Keywords:
Visual representations learning
Action recognition
Self-supervised learning

ABSTRACT

Large labeled datasets are crucial for video understanding progress. However, the labeling process is time-consuming, expensive, and tiresome. To overcome this impediment, various pretexts use the temporal coherence in videos to learn visual representations in a self-supervised manner. However, these pretexts (order verification and sequence sorting) struggle when encountering cyclic actions due to the label ambiguity problem. To overcome these limitations, we present a novel temporal pretext task to address self-supervised learning of visual representations from unlabeled videos. Repeated Scene Localization (RSL) is a multi-class classification pretext that involves changing the temporal order of the frames in a video by repeating a scene. Then, the network is trained to identify the modified video, localize the location of the repeated scene, and identify the unmodified original videos that do not have repeated scenes. We evaluated the proposed pretext on two benchmark datasets, UCF-101 and HMDB-51. The experimental results show that the proposed pretext achieves state-of-the-art results in action recognition and video retrieval tasks. In action recognition, our S3D model achieves 88.15% and 56.86% on UCF-101 and HMDB-51, respectively. It outperforms the current state-of-the-art by 1.05% and 3.26%. Our R(2+1)D-Adjacent model achieves 83.52% and 54.50% on UCF-101 and HMDB-51, respectively. It outperforms the single pretext tasks by 8.7% and 13.9%. In video retrieval, our R(2+1)D-Offset model outperforms the single pretext tasks by 4.68% and 1.1% Top 1 accuracies on UCF-101 and HMDB-51, respectively. The source code and the trained models are publicly available at https://github.com/Hussein-A-Hassan/RSL-Pretext.

1. Introduction

Video understanding requires large, labeled datasets for supervised training of large-scale video models, which are essential for daily applications such as autonomous driving, surveillance and security. Many large datasets are created, such as Moments in Time, Kinetics-600, and Something-Something. Creating these datasets requires a considerably long time and effort due to the time-consuming, laborious, and costly nature of annotation. In addition, millions of unlabeled videos on the internet can be used to unshackle advances in video understanding. Therefore, self-supervised learning has significant importance since it uses the freely available supervisory signals derived from the data to learn valuable representations. Self-supervised methods can be divided into contrastive learning methods and pretext methods. Self-supervised contrastive methods require more computational resources than pretext tasks, since they rely on contrasting many positive pairs with many negative pairs in each batch. In addition, some contrastive approaches require a large batch size to increase the number of negative pairs. Others rely on storing a queue of representations from previous batches to use as negatives.

Several pretexts use temporal coherence, temporal order, in videos as a supervisory signal. Examples of such pretexts include temporal order verification, sorting temporally shuffled frames or clips, and the arrow of time prediction. These order verification and sequence sorting pretexts struggle to overcome the ambiguity of cyclic actions in which two different sequence orders of the same video are valid and possible. Examples of cyclic actions include opening/closing a door [1], picking up/placing down a coffee cup [2], pull-up [3], and swinging a child [2]. This ambiguity exists because cyclic actions do not have unique clips/frames order since both forward and backward playbacks are correct and valid. Fig. 1 shows an example of a cyclic action, juggling balls, which has a repetitive, and rhythmic motion since the balls go up and come down in a continuous loop. Clip A represents the natural temporal order of the action of juggling balls, while clip B represents the backward playback temporal order. Clips A and B clearly

* Corresponding author.
E-mail addresses: hussein@mu.edu.iq (H. Altabrawee), halimnoor@usm.my (M.H. Mohd Noor).

https://doi.org/10.1016/j.patcog.2024.110804
Received 3 February 2022; Received in revised form 27 April 2024; Accepted 15 July 2024
Available online 18 July 2024

Fig. 1. Cyclic actions ambiguity. Clips A and B are valid and correct, leading to label ambiguity. The RSL finds an anomaly in the video (the scenes in the red boxes), which does not depend on the temporal order; therefore, it is not affected by the ambiguity. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

show that both forward and backward temporal orders are correct and valid, which represents a challenge to the order verification and sequence sorting pretexts because of the label ambiguity problem.

To overcome these limitations, we propose a novel temporal pretext unaffected by the cyclic actions ambiguity because it does not depend on finding unique clips/frames order. Instead, our pretext depends on finding spatial and temporal change, an anomaly in the video coherence that is duplicated. Whether the sequence is played using forward or backward playback does not matter, as represented in clips C and D in Fig. 1. The key idea of the pretext task is modifying the frames' temporal order in a video by repeating a scene (represented by the red boxes in Fig. 1). Then, a neural network is trained to identify the modified video, localize the location of the repeated scene, and identify the unmodified original videos that do not have repeated scenes. We evaluated the proposed pretext task, Repeated Scene Localization (RSL), on action recognition and video retrieval tasks using two benchmark datasets, UCF-101 and HMDB-51. The experimental results show that the proposed RSL pretext achieves state-of-the-art performance with only RGB modality as input. The contributions of this research are as follows:

- A novel pretext, Repeated Scene Localization (RSL), is proposed. Our pretext is unaffected by cyclic actions' ambiguity and requires much less memory than self-supervised contrastive methods.
- The RSL pretext is evaluated on two benchmark datasets, UCF-101 and HMDB-51. It achieves state-of-the-art performance in action recognition and video retrieval. In addition, ablation experiments are performed, showing its performance under different settings.

2. Related work

Action Recognition: Action recognition is a dynamic and active area of research. [4] introduced a method called Temporal Segment Dropout (TSD) as a form of temporal regularization. TSD prioritizes enhancing the temporal representations in a clip of temporal segments by disregarding the most prominent spatial representations. [5] proposed a Hybrid Attention-guided ConvNeXt-GRU Network for enhancing action recognition accuracy. The approach relies on incorporating a multi-scale hybrid attention module combined with GRU into ConvNeXt to dynamically adjust channel features and extract both global and local information from various regions. [6] employed causal inference to intervene on the action to eliminate the domain background's confounding effect on the class label to achieve video domain generalization. [7] presented a novel model design named Convolutional Transformer Network that combines the advantages of CNNs with the advantages of the Transformer. The architecture involves the implementation of a video-to-tokens module that creates tokens from the videos' spatial–temporal representations produced by 3D convolutions. [8] introduced an approach of sequences of transformer encoders to minimize the computational complexity of the Vision Transformer by utilizing a relative-position embeddings scheme instead of the absolute-position embeddings.

Video Retrieval: Recent research [9] introduced a framework for content-based video retrieval that eliminates frames that are easily recognized as distractions and estimates the extent to whether the remaining frames must be eliminated, taking into account saliency information and subject relevancy. [10] presented a new composed video retrieval method that employs detailed language descriptions for explicit encoding query-specific contextual data. The method learns discriminative embeddings of vision-only, text-only, and vision-text to improve the alignment and precisely retrieve corresponding target videos. [11] presented a video retrieval approach, which integrates attention technique and multimodal fusion using the SlowFast backbone. The slow network is responsible for obtaining skeleton motion representations, while the fast network is responsible for obtaining static image representations from video sequences. [12] presented a video-text retrieval approach that utilizes an Adaptive Attribute-Aware Graph Convolutional Network and a Multi-Modal Masked Transformer. The approach uses an adaptive correlation matrix in order to acquire distinctive video features for video-text retrieval. They presented a new loss function to enhance the accuracy of measuring the semantic similarity between video-text pairs during the training. [13] introduced a dual inter-modal interaction framework called DI-VTR, enabling video-text retrieval. The framework incorporates a module for dual inter-modal interaction to achieve precise multilingual alignment across both the text and video modalities.

Self-supervised Learning: Recently, researchers proposed a contrastive learning approach to make the learned features along the temporal dimension more distinct. The approach uses two novel contrastive losses. One loss discriminates between two distinct non-overlapping clips sampled from the same video. The second loss discriminates among the time-steps of the feature map taken from a clip [14]. Another method combines clustering and contrastive learning for learning visual representations. The positives and negatives are constructed based on the cluster assignments. The positives come from the same cluster, while the negatives are all other representations from other clusters [15].

Changing the temporal dimension of the video to get a supervision signal is a typical pretext design technique. Temporal order verification is a binary classification task that determines whether three video frames in a tuple are in the correct temporal order or not using 2D-CNN [2]. The frames are sampled from high-motion windows that are identified using optical flow. Sorting video frames is a multi-class classification pretext for determining the correct temporal order of randomly shuffled video frames using 2D-CNN [1]. The frames are sampled based on the motion magnitude, which is calculated using optical flow. The forward and backward permutations for a video are considered one class to overcome the ambiguity of cyclic actions. However, this limits the supervisory signal for non-cyclic actions since there would be no difference between forward and backward playbacks. Finding the wrong video subsequence in a tuple of subsequences is a multi-class classification task that identifies the subsequence that has incorrect frames' temporal order using 2D-CNN [16]. Each subsequence is encoded to a 2D image or stack of differences of frames, which is not optimal for representing the temporal dynamics in videos and requires preprocessing computations. These methods use computationally expensive optical flow to find high-motion segments or motion magnitudes for sampling frames. In addition, they use 2D-CNN encoders that cannot model temporal information. Unlike these approaches, RSL

is not affected by the ambiguity of cyclic actions, and it does not use optical flow or rely on 2D-CNN.

[3] proposed future frames order ranking based on a context clip, which is used to solve the ambiguity of sorting frames/clips without context. The method samples the context clip and then samples many target frames from the video segment that follows it. It measures the ranking scores between each of the target features relative to the context features. In [3], the pretext ranking objective learns only temporal content and neglects the spatial content; therefore, they added a contrastive objective. However, optimizing both objectives makes this method complex and depends on the size of the negative samples. In addition, it does not generalize well on the downstream tasks, and they added an extra auxiliary pretext (rotation prediction). Unlike this approach, RSL is straightforward and does not rely on using negative samples or multi-objective optimization.

[17] presented sorting a tuple of shuffled clips as a temporal pretext using 3D-CNN. The network has to predict the correct order of the clips. [18] proposed detecting wrong clips, which have incorrect frames' temporal order. The pretext creates wrong clips using predefined temporal permutations, such as random permutation, split and swap, and play backward. The 3D-CNN has to identify the location of the wrong clip among many normal clips. [19] proposed predicting the playback direction as a pretext, which is a binary classification task, using optical flow as input. [20] proposed predicting the type of temporal transformation applied to video clips as a pretext, which is a multitask pretext that identifies the speed of a clip and the type of temporal transformation applied to it using 3D-CNN. This method uses four different speeds and four different temporal transformations. These transformations include speed, random permutation, periodic, and warp, which uses a random sub-sampling selection of frames. One transformation relies on the playback direction, which makes it affected by the ambiguity of cyclic actions.

Although the aforementioned approaches have established being able to learn visual representations from unlabeled videos, some methods rely on 2D-CNN, which is not effective in modeling the temporal dimension of the videos [1,2,16]. Another limitation is using optical flow as the input modality, which is computationally expensive [19]. In addition, these temporal pretexts suffer from cyclic actions' ambiguity because they rely on the playback direction of the input videos.

Repeating video frames has been used in the literature. [21] proposed a method that repeats an entire clip in the reverse temporal order, then concatenates the two clips to create one palindrome. [22] proposed a method that repeats a single frame multiple times to create a static clip without motion, which is used as an intra-negative clip. Unlike these methods, RSL uses a scene with motion to create a repeated scene in the clip, without repeating the entire clip or reversing the temporal order of the repeated frames.

The RSL's objective is twofold. The first objective involves distinguishing between unmodified clips and those with repeated scenes, while the second involves identifying the repeated scene within the clip. This goal necessitates comparing all the scenes in the temporal dimension. Unlike the previously proposed pretexts that modify the temporal order by rearranging and permuting the frames randomly or in a predefined order, RSL uses the frame repeating to repeat a scene in the clip, which changes the order of the frames.

3. Repeated scene localization pretext task

Humans possess an inherent capacity to discern whether an action or a scene from an action has been previously observed and occurred in the past. The determination of whether a scene occurred is contingent upon our comprehension of action dynamics and our recollection. Amazingly, humans notice repeated actions instantly. We hypothesize that a deep model trained to localize a repeated scene in a video clip will learn useful representations for downstream tasks. To solve the RSL pretext, the model must examine and compare the sequence of scenes in the video. Fig. 2 shows the RSL's dynamic aspect of execution and the downstream task evaluation from beginning to end. The input clip generator creates the label for each clip automatically based on the existence of a repeated scene in the clip and the location of the scene. For example, an input clip with a repeated scene starting from the frame at index four will have the label four, while an input clip that does not have a repeated scene will have the label None.

To create a clip with a repeated scene, we sample fixed-length clips with forward playback starting at random locations from the original videos at each epoch, as depicted in Fig. 2 by the activities performed by the clip sampler. The length of the clips, l, and the sampling speed, s, are determined based on each experiment. Let n represent the number of unique frames in the clip and r the number of frames to repeat (repeated scene length). Let $V = (f_1 \ldots f_{n+r})$ represent a clip, sampled at a specific speed, starting at a random frame from an original video. It has $n+r$ temporally arranged frames. For example, $n+r$ could be 16, $n = 12$, and $r = 4$. The procedure for creating a clip with a repeated scene of length r is as follows:

- Select n consecutive frames from V starting at a random location to represent the first segment, which contains unique frames, of the new clip. Fig. 2 illustrates this after the input clip generator determines to create a modified clip with a repeated scene.
- Select r consecutive frames from the first segment to be repeated. The selection starting index is chosen randomly from a fixed list of indices representing specific predefined positions. The chosen index will be used as a label for the generated clip. These frames represent the second segment of the newly generated clip. Fig. 2 illustrates this by the activities performed by the scene selector.
- Select the insertion index, joining location, based on the experiment settings. Fig. 2 illustrates this by the activities performed by the insertion index finder.
- Join the two segments to create a clip that has a repeated scene. Fig. 2 illustrates this by the input clip generator activity after receiving the insertion index.

The repeated scene location and the insertion location determine the temporal arrangement of the frames in the new clip, as illustrated in Fig. 3. For instance, configuration 1 has few temporal changes because the repeated scene is adjacent to the insertion location, while configurations 2 and 3 have more temporal changes as the repeated scene is far away from the insertion location.

An effective pretext should have the right difficulty level to challenge the network to learn useful representations. If the pretext is ambiguous, the network cannot solve the task and will not learn the visual representation. Two parameters determine the RSL's difficulty. The first is r, the number of repeated frames. When r is small compared to the clip length, the task will be more challenging. For example, repeating two frames in a clip of 32 frames is more challenging to solve than repeating four frames in a clip of 16 frames. The second is the insertion location of the replicated scene, which can be either fixed or random, thereby determining the number of distinct clips that can be generated from the original clip. When using fixed insertion, the task will be easier to solve because the number of different possible generated clips will be small. In contrast, when using random insertion, the number of possible clips generated will be large. Consequently, the task will be harder to solve.

Fig. 4 illustrates an example of the RSL pretext. The example uses $l = 16$, $n = 12$, and $r = 4$. For each clip, there are four labels, [0, 4, 8, None], which will be encoded in the implementation as [0, 1, 2, 3], respectively. First, a clip is sampled from the original video at a random sampling speed, s, from a random starting frame. We skip three frames in this example ($s = 4$), and the sampled clip starts at frame ten from the original video. Then, the clip will be randomly selected to be modified by repeating a scene or not. If the clip is not modified, then the label will be None, and the clip will be fed to the 3D-CNN.

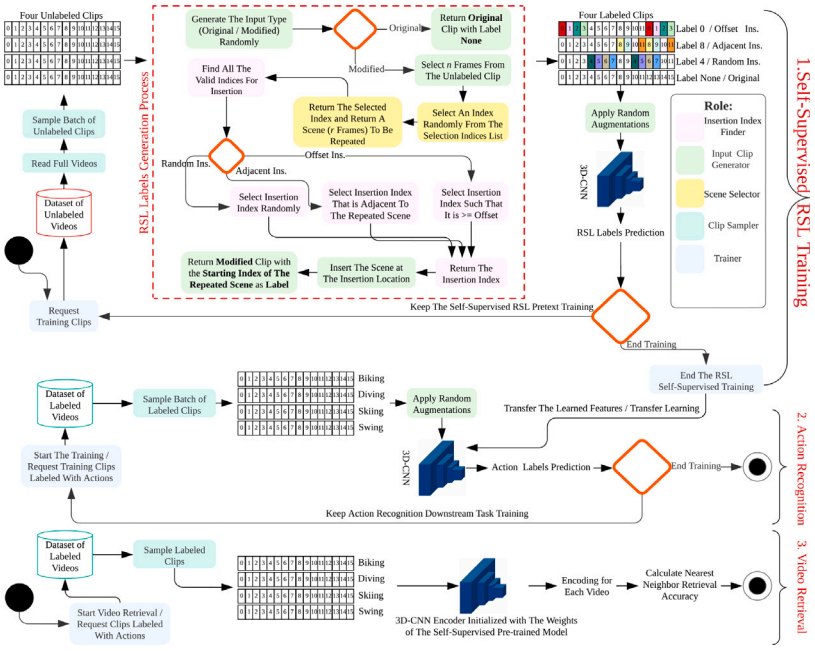

Fig. 2. RSL activity diagram that shows the dynamic aspect of execution RSL pretext and the downstream tasks evaluation.

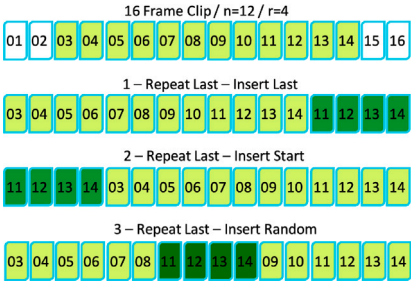

Fig. 3. Different configurations of repeating a scene.

If the clip is selected to be modified, a frame index is randomly selected from a predefined selection indices list, where in the example, the list is [0, 4, 8]. The selected index will be the label of the sampled clip; in this case, the label is 0 (frame index 0 is selected). This label means that the repeated scene will start at the first frame of the first segment created in step 3. Then, the repeated scene is inserted into the first segment to create a clip with a repeated scene. Finally, the generated clip is used as input to the 3D-CNN for the self-supervised RSL training with four labels: [0, 4, 8, None]. The training is performed by minimizing the cross-entropy loss. The red, green, and gray bars represent the predicted class probabilities.

4. Experimental results

4.1. Datasets

We used UCF-101 [23] and HMDB-51 [24] datasets for action recognition and video retrieval downstream tasks, while we used the training set of UCF-101 split 1 for self-supervised training. UCF-101 has 13,320 diverse and realistic videos of human actions downloaded from YouTube. HMDB-51 has 6849 videos that cover 51 human actions. We used the official splits of the datasets and followed the most common protocol for self-supervised pretext training and downstream task evaluation on action recognition and video retrieval.

4.2. Ablation study

We performed ablation experiments to illustrate the impact of altering the RSL difficulty by varying the number of labels (2, 4, and 8) through adjustments to the n and r hyperparameters and to illustrate the impact of altering the insertion mode.

Self-supervised Pre-training: The default self-supervised pre-training settings include using the training set of UCF-101 split 1 for the self-supervised RSL pre-training, without employing any action labels. We used the official R(2+1)D PyTorch implementation with a clip size of 16 × 112 × 112. To create the input clips, we resized the clips to a resolution of 16 × 128 × 171 and, then used a random crop of 16 × 112 × 112. We sampled the input clips at a random speed between 1 and 4. We used a batch size of 16 clips, the SGD optimizer, and a reduce-on-plateau scheduler with patience equal to 20 epochs. We added two linear layers to the end of the network with a dropout rate of 0.6. To prevent the models from using shortcuts, we used spatial inconsistent data augmentations applied gradually during the pre-training. They include a random-sized crop, a random horizontal flip, a random gray, a random color jitter, and a random Gaussian blur. These are our default pre-training settings, any deviations from these settings are specified throughout the paper.

We followed the aforementioned default settings for all experiments. In the ablation pre-training, we trained the R(2+1)D models for 244 epochs with an initial learning rate of $1e-3$, a weight decay of $1e-3$, and a momentum of $9e-1$. Row 2 of Table 1 presents the different values of the n and r hyperparameters and the number of labels in each experiment. We executed three experiments where we inserted the repeated scene such that there was an offset of frames between it and its replica. This insertion mode, offset, mimics configuration 2 in Fig. 3, ensuring a gap between the two scenes.

Downstream Task Evaluation: The default downstream task evaluation includes using action recognition and video retrieval to evaluate the effectiveness of the RSL pretext. We used all three splits of UCF-101 and HMDB-51 for action recognition fine-tuning. We used the self-supervised RSL pre-trained model as weight initialization by removing the RSL classification head and attaching a new randomly initialized classification head on top of the model. Then all the layers are fine-tuned with cross-entropy loss. We added two linear layers to the top of the network with a dropout rate of 0.9. We used one cycle learning

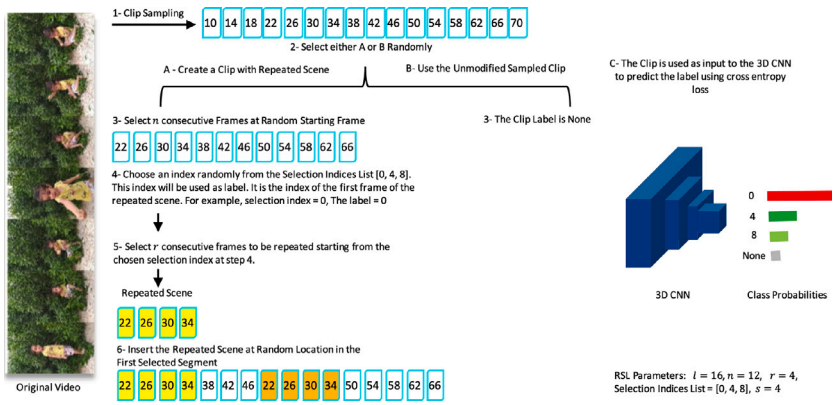

Fig. 4. RSL labels generation. Creating a 16 frames clip with four repeated frames and four possible labels. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 1
Ablation experiments. The effect of different configurations on RSL pretext.

Exp.	R	N	Label	Ins. mode	RSL Acc.	Top1	Top5	Top10	Top20	Top50	Action Recog.
Rand.	–	–	–	–	–	–	–	–	–	–	72.13
1	2	14	8	Offset	65.5	26.9	43.7	53.5	62.2	74	**83.76**
2	4	12	4	Offset	88.76	**27.3**	**45.7**	**54.4**	**63.6**	**75.4**	83.1
3	8	8	2	Offset	**95.08**	26.2	44.7	54	63.1	75	83.69
4	4	12	4	Random	75.49	25	42.7	52	61.7	73.5	83.66
5	4	12	4	Adjacent	84.69	26.6	43.8	52.7	62.2	74.7	83.74

rate scheduler, a batch size of 16, and the SGD optimizer. For R(2+1)D models, we resized the clips to a resolution of $16 \times 128 \times 171$ then used a random crop of $16 \times 112 \times 112$ to create the input clips. We used a skip rate of three frames (a speed of 4) to sample the input clips, a random color jittering, a random horizontal flip, and a random crop as augmentations during training. For testing, we resized the input clips to $16 \times 128 \times 171$ and used a center crop of $16 \times 112 \times 112$.

In the video retrieval task, we used split 1 of each dataset for the evaluation. In each dataset, we used the clips in the test set of split 1 to query the clips in the training set of split 1 following [17,25]. We did not train or fine-tune a video retrieval model following [17,25] by using the self-supervised pre-trained network, R(2+1)D, as a feature extractor. To calculate the video-level features, we sampled ten clips from each video, and their features were extracted using the self-supervised pre-trained model. Max pooling was applied to the feature maps of the penultimate layer (prior to the last layer) instead of global spatiotemporal pooling to extract the features. The clips in the test set were used to query the training set clips. The cosine distances between a query clip's features and all the clips' features in the training set were computed. If the label of the test clip was found in the labels of the k nearest training clips, then it was regarded as correctly predicted.

During action recognition inference, the video-level prediction was calculated by averaging the predictions of 10 center-cropped clips sampled uniformly from the video following prior works [17,25]. Features for the nearest-neighbor retrieval experiments were similarly calculated by averaging features from uniformly sampled ten center crops following prior works [17,25]. These are our default downstream task evaluation settings, any deviations from these settings are specified throughout the paper.

We followed the aforementioned default settings for all ablation experiments, with the following exceptions: Only the training set of UCF-101 split 1 is used to fine-tune the models, while the testing set of UCF-101 split 1 is used to test the action recognition models. We fine-tuned the models, R(2+1)D, for 360 epochs, with an initial learning rate of $2e-3$, a weight decay of $1e-3$, and a momentum of $9e-1$. Only UCF-101 split 1 is used as the validation set in the video retrieval ablation evaluation. Table 1 shows our pretext performance on the action recognition and video retrieval for these ablation experiments.

Row 2 of Table 1 shows that the RSL difficulty level is controlled by the n and r hyperparameters. Experiment 1, which has eight labels, is the most difficult pretext, with an accuracy of 65.5%. Experiment 2, which has four labels, is less difficult since its accuracy is 88.76%, while Experiment 3 is the easiest since it is a binary classification with only two labels with an accuracy of 95.08%. All three experiments achieve very similar fine-tuning accuracy for action recognition, as the difference between their scores is less than 1%. All the experiments show the ability of the RSL pretext to learn excellent and useful representations used for action recognition. Compared to the randomly initialized model, the RSL self-supervised models achieve at least a 10% increase in action recognition accuracy. For video retrieval, all the models achieve comparable scores, yet the scores of Experiment 2 are slightly better than the other two experiments. Based on these results, the labels are set to four with $n = 12$ and $r = 4$ for all our other experiments.

We conducted another two ablation experiments. These experiments show the effects of using different configurations for inserting the repeated scene. The random insertion mode mimics configuration 3 in Fig. 3, which allows the repeated scene replica to be inserted at a random location close to or far away from the original scene. In contrast, the adjacent insertion mode mimics configuration 1 in Fig. 3, ensuring that the repeated scene replica is inserted at an adjacent location to the repeated scene. We used the same previous self-supervised RSL training procedure and downstream task evaluation procedure in these two experiments. Row 3 in Table 1 shows the RSL performance on action recognition and video retrieval for these ablation experiments.

Row 3 in Table 1 shows that the insertion mode affects the RSL difficulty level. Experiment 4, which uses random insertion, is the most difficult pretext, with an accuracy of 75.49%. Experiment 5, which uses adjacent insertion mode, is less difficult since its accuracy is 84.69%, while Experiment 2, which uses an offset insertion, is the easiest since its accuracy is 88.76%. All the experiments have very similar fine-tuning accuracy for action recognition, since the difference between their accuracies is less than 1%. All the experiments prove the ability of the RSL pretext to learn useful representations for action recognition. Compared to the randomly initialized model, the RSL self-supervised models achieve at least a 10% increase in action recognition accuracy.

Fig. 5. R(2+1)D-Adjacent model's attention maps.

For video retrieval, the scores of Experiment 2 are better than the other two experiments. Fig. 5 shows the attention maps of the top three predictions for two actions using the R(2+1)D-Adjacent model. Clearly, our model focuses on the areas and objects related to the action.

4.3. The RSL computational cost and efficiency

Computational Complexity: The core operations of the RSL pretext are the selection and insertion operations. The complexity of the selection operation is constant, selecting an index randomly from a predefined list of indices. Three modes (offset, random, and adjacent) are used to execute the insertion operation. The computational complexities of the random and adjacent insertions are constant, selecting an index randomly from a predefined list of insertion indices. However, the offset insertion requires that the insertion index be far away by an appropriate offset of frames from the scene to be repeated. This condition has to be checked for every insertion index candidate; therefore, the worst-case computational complexity of the offset insertion is $O(n)$, which represents the case of searching all the video frames for a valid insertion index.

The order verification and sequence sorting pretexts have constant computational complexity since they depend on changing the clips/frames order based on predefined permutations. However, this complexity does not apply to the methods that require preprocessing, such as finding high-motion segments using optical flow [2], finding motion magnitude [1], or preprocessing the clips to produce other modalities [16].

Training and Inference Time: Training the R(2+1)D model for 244 epochs to solve the RSL pretext on the full training set of UCF-101 split 1, which has 9537 videos, takes approximately two days on a single RTX 3080 10 GB GPU, with Intel 10700K CPU. It takes approximately 10.48 min to train the model for one epoch. The inference time on the full testing set of UCF-101 split 1, which has 3783 videos, takes approximately 2.38 min.

4.4. RSL pre-training

We used two models, R(2+1)D and S3D, to solve the RSL pretext. To generate the labels for R(2+1)D and S3D models, we used (0, 4, 8, None) and (0, 16, 32, None), respectively. We trained three R(2+1)D models (R(2+1)D-Random, R(2+1)D-Adjacent, R(2+1)D-Offset) using random, adjacent, and offset insertion modes, respectively. Our default pre-training settings, as stated in Section 4.2, were followed. We used four RSL labels, r equals four, a learning rate of 1e–3, a weight decay of 1e–3, and a momentum of 9e–1 for all three models. R(2+1)D-Random and R(2+1)D-Offset were trained for 400 epochs, while R(2+1)D-Adjacent was trained for 244 epochs. For the S3D model, we followed our default pre-training settings mentioned in Section 4.2, with the following exceptions. We used random insertion mode, clips of 64 frames, four RSL labels, r equals 16, 200 training epochs, a batch size of 5, a learning rate of 1e–3, a weight decay of 1e–5, and a momentum of 9e–1. We resized the clips to a resolution of 64 × 256 × 256, then a random crop of 64 × 224 × 224 was used.

4.5. Action recognition

We followed our default fine-tuning and evaluation settings mentioned in Section 4.2 for the action recognition downstream task. R(2+1)D-Random was fine-tuned on UCF-101 and HMDB-51 using a learning rate of 1e–2, a weight decay of 1e–3, and a momentum of 9e–1. We fine-tuned R(2+1)D-Random for 477 epochs on UCF-101 three splits, while we fine-tuned it for 541, 496, and 433 on HMDB-51 three splits, respectively. R(2+1)D-Adjacent was fine-tuned on UCF-101 and HMDB-51 using a learning rate of 2e–3, a weight decay of 1e–3, and a momentum of 9e–1. We fine-tuned R(2+1)D-Adjacent for 431 and 400 epochs on UCF-101 three splits and HMDB-51 three splits, respectively. For the S3D-Random model, we followed our default fine-tuning and evaluation settings mentioned in Section 4.2, with the following exceptions. We used only split 1 of UCF-101 and HMDB-51 for fine-tuning and testing following [25]. On both UCF-101 and HMDB-51, we used clips of 64 frames, a sampling rate of 1, a batch size of 32, a momentum of 0.99, and a reduce-on-plateau scheduler with patience of 20 epochs. The clip was resized to 64 × 256 × 256, and a color jittering, a random horizontal flip, and a random crop to 64 × 224 × 224 were used as augmentations during training. We resized the clip to 64 × 256 × 256, and a center crop of 64 × 224 × 224 was used for testing. The S3D-Random model was fine-tuned for 201 and 200 epochs on UCF-101 and HMDB-51, respectively. We used a learning rate of 1e–3 and a weight decay of 1e–3 on UCF-101, while we used a learning rate of 4e–3 and a weight decay of 4e–5 on HMDB-51. The weight decay and the learning rate decreased gradually during training.

For the R(2+1)D network, we report the average Top 1 classification accuracy over the three splits of UCF-101 and HMDB-51. For the S3D network, we report the Top 1 classification accuracy of split 1 only of UCF-101 and HMDB-51 following [25]. In order to make a fair comparison, we compare our work with the self-supervised approaches that use a single pretext trained on UCF-101 using the RGB modality only. Optimizing and adding other losses, such as multitask pretext and contrastive loss, could boost the accuracy. However, it will require much more computational resources and training time than our proposed pretext. It is noteworthy that an entirely fair comparison between all the methods in the literature is difficult because these methods use different networks, sampling rates, resolution, clip length, and video level evaluation procedures. For example, various techniques are used to calculate the video level prediction, such as using ten center-cropped clips sampled uniformly, ten crops with ten clips each sampled uniformly, or all the center-cropped clips in the video. For reference, we list the performance of those more complex multitask contrastive pretexts in Table 2. Fig. 6 shows the attention maps of the top three

Table 2
Action recognition. All networks are self-supervised pre-trained on UCF-101 Split 1 training set using the RGB modality only.

Method	Network	Resolution	Frames	UCF	HMDB
Single pretexts					
Video Order [18]	3D-AlexNet	–	8	49.2	–
Shuffle and Learn [2]	CaffeNet	227 × 227	–	50.2	18.1
Skip-Clip [3]	3D ResNet-18	112 × 112	16	59.5	–
OPN [1]	VGG	80 × 80	–	59.8	23.8
DPC [26]	3D-ResNet18	128 × 128	25	60.6	–
VCP [27]	C3D	112 × 112	16	68.5	32.5
MemDPC [28]	ResNet18+RNN	128 × 128	40	69.2	–
VCOP [17]	R(2 + 1)D	112 × 112	16	72.4	30.9
VTDL [29]	C3D	112 × 112	16	73.2	40.6
Pace [25]	R(2 + 1)D	112 × 112	16	73.9	33.8
PSP [30]	R(2 + 1)D	112 × 112	16	74.82	36.82
RSL (Ours) - R(2 + 1)D-Random		112 × 112	16	80.68	52.09
RSL (Ours) - R(2 + 1)D-Adjacent		112 × 112	16	83.52	54.50
RSL (Ours) - S3D-Random		224 × 224	64	**88.15**	**56.86**
Multitask pretexts - Contrastive					
Skip-Clip-Aux [3]	3D ResNet-18	112 × 112	16	64.4	–
PRP [31]	R(2 + 1)D	112 × 112	16	72.1	35.0
Pace-Contrastive [25]	R(2 + 1)D	112 × 112	16	75.9	35.9
V3S [32]	R(2 + 1)D	112 × 112	16	79.1	38.7
Temporal Trans. [20]	R(2 + 1)D	112 × 112	16	81.6	46.4
Vi2CLR [15]	S3D	128 × 128	32	82.8	52.9
TCLR [14]	R(2 + 1)D	112 × 112	16	82.8	53.6
TCLR [14]	R3D-18	112 × 112	16	83.9	53.5
V3S [32]	S3D-G	224 × 224	64	85.4	53.2
Pace-Contrastive [25]	S3D-G	224 × 224	64	87.1	52.6

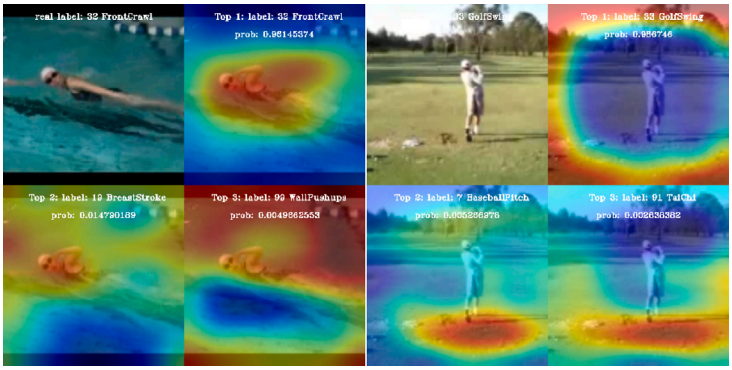

Fig. 6. R(2+1)D-Random model's attention maps.

predictions for two actions using the R(2+1)D-Random model. Clearly, our model focuses on the areas and objects related to the action.

Our proposed pretext achieves state-of-the-art performance compared to the previously proposed single pretexts. It outperforms all the other methods by a large margin, 83.52% compared to 74.82% on UCF-101 and 54.50% compared to 40.6% on HMDB-51 for the R(2+1)D-Adjacent network. Our method is higher by 8.7% and 13.9%, respectively.

When we compare our R(2+1)D-Adjacent network with the more advanced multitask contrastive self-supervised approaches, our method outperforms the R(2+1)D model proposed by TCLR [14] by 0.72% and 0.9% on UCF-101 and HMDB-51, respectively. In addition, our model outperforms the R3D-18 model proposed by TCLR [14] by 1% on HMDB-51 and achieves comparable performance on UCF-101. Our RSL pretext achieves outstanding performance compared to these methods, which use much more complex multi-loss functions requiring higher computational cost and training than our pretext. Our R(2+1)D-Adjacent model achieves higher accuracy than our R(2+1)D-Random model, even though they both achieve comparable results in the ablation experiments. One reason could be that the fine-tuning hyperparameters used in the R(2+1)D-Adjacent fine-tuning training are more suitable than those used in R(2+1)D-Random training.

Our S3D-Random model achieves state-of-the-art accuracy, 88.15% and 56.86% on UCF-101 and HMDB-51 compared to the multitask contrastive pretext proposed by [25]. The accuracy of our S3D-Random model is higher than [25] by 1.05% and 4.26% on UCF-101 and HMDB-51, respectively. Our S3D-Random model outperforms the TCLR R(2+1)D model [14] by 3.26% on HMDB-51. Fig. 7 shows the attention maps of the top three predictions for two actions using the S3D-Random model. Clearly, our model focuses on the areas and objects related to the action.

4.6. Video retrieval

We followed our default video retrieval evaluation procedure mentioned in Section 4.2. We report the retrieval scores for two models, R(2+1)D-Offset and R(2+1)D-Random. Table 3 shows our pretext performance on the video retrieval task.

Our pretext achieves state-of-the-art when compared to the single pretexts, top row in the table, even though some pretexts used different backbones such as (2+3D)-ResNet18 in [28] and R3D in [30]. For UCF-101, our pretext (R(2+1)D-Random model) is higher by 4.7%, 5.3%, 3.7%, and 0.6% than the single pretexts for Top 1%, Top 5%, Top 10%, and Top 20% respectively. Our pretext is less than [30] by only

Fig. 7. S3D-Random model's attention maps.

Table 3
RSL video retrieval performance on UCF-101/HMDB-51.

Method	Top1%	Top5%	Top10%	Top20%	Top50%
VCOP [17]	10.7/5.7	25.9/19.5	35.4/30.7	47.3/45.8	63.9/67.0
VCP [27]	19.9/6.7	33.7/21.3	42.0/32.7	50.5/49.2	64.4/73.3
MemDPC [28]	20.2/7.7	40.4/25.7	52.4/40.6	64.7/57.7	–/–
PSP [30] (R3D)	24.6/10.3	41.9/26.6	51.3/38.8	62.7/54.6	76.9/76.8
RSL R(2 + 1)D-Rand.	29.3/11.1	47.2/29.9	56.1/43.4	65.3/58.0	76.3/77.7
RSL R(2 + 1)D-Offset	29.28/11.4	47.36/30.9	56.04/43.9	64.55/58.8	76.31/77
Multitask pretexts					
PRP [31]	20.3/8.2	34.0/25.3	41.9/36.2	51.7/51.0	64.2/73.0
Temporal Trans. [20]	26.1/–	48.5/–	59.1/–	69.6/–	**82.8**/–
V3S [32]	23.1/9.6	40.5/24.0	48.7/37.2	58.5/54.3	72.4/**77.9**
Pace-Contrastive [25]	25.6/12.9	42.7/31.6	51.3/43.2	61.3/58.0	74.0/77.1
Vi2CLR [15] S3D	55.4/**24.6**	70.9/45.1	78.3/54.9	83.6/67.6	–/–
TCLR [14]	**56.9**/24.1	**72.2**/**45.8**	**79.0**/**58.3**	**84.6**/**75.3**	–/–

0.6% for the Top 50% accuracy. For HMDB-51, our pretext (R(2+1)D-Offset model) is higher by 1.1%, 4.3%, 3.3%, 1.1%, and 0.2% than the single pretexts for Top 1%, Top 5%, Top 10%, Top 20%, and Top 50% respectively. When compared to the contrastive and multitask methods, our pretext achieved competitive performance compared to some methods, while the recent methods, Vi2CLR and TCLR, achieved an impressive performance that outperforms all the pretexts by a large margin. These methods optimize much complex multi-loss functions.

5. Qualitative analysis

Fig. 8 shows a qualitative analysis of our RSL pretext. We compare our R(2+1)D-Adjacent model with a randomly initialized R(2+1)D model on the video retrieval task. In the figure, each video clip is represented by two frames. For each testing video, the two nearest neighbors were retrieved from the training set of UCF-101 split 1. Row A shows the retrieved videos of the randomly initialized R(2+1)D model, while Row B shows the retrieved videos of our model. Our model focuses on motion dynamics. For instance, it successfully retrieves two videos of the BandMarching class. These videos are visually different, yet their motion dynamics are similar.

6. Limitations

Our pretext uses an unlabeled clip sampled from a random location in the original video. In addition, it uses fixed positions for selecting a scene to be repeated. There is no guarantee that the sampled clip or the selected scene contain high-motion information, which is crucial for learning actions. Sampling a clip or selecting a scene that has a high-motion signal could increase the quality of the representations. Other than the costly optical flow, one modality that contains noisy motion signals is motion vectors, which are already available through video codecs such as MPEG-4 and H.264. Using motion vectors to sample a clip or select a scene with high motion could increase our pretext performance. In addition, video frame differences is another modality that could be used as input. Using frame differences could encode some motion aspects of the video. Moreover, exploring untrimmed videos or larger datasets like Kinetics can be an intriguing endeavor. In addition, our RSL pretext, like several other pretexts, achieves lower video retrieval accuracies when compared with contrastive approaches. The reason could be that the pretexts indirectly encourage the model to learn visual representations via a proxy task. In contrast, contrastive methods explicitly and directly try to group or cluster the representations of similar videos together in the representation space via contrastive losses.

7. Conclusions

We presented a novel self-supervised pretext used to learn visual representations from unlabeled videos. Our RSL pretext is straightforward, unaffected by the ambiguity of cyclic actions, efficient in training, and does not require a large memory or computational resources. In action recognition, our S3D model achieves 88.15% and 56.86% on UCF-101 and HMDB-51, respectively. These scores are higher than the state-of-the-art by 1.05% and 3.26%. In addition, our R(2+1)D model achieves 83.52% and 54.50% on UCF-101 and HMDB-51, respectively. These scores are 8.7% and 13.9% higher than the accuracies of single pretexts. In addition, we achieve 4.68% and 1.1% gain in the Top 1 video retrieval accuracies on UCF-101 and HMDB-51, respectively. Our ablation experiments show that the RSL pretext is very effective in learning visual representations under different difficulty levels. Our findings prove that the utilization of these representations has the potential to enhance the accuracy of action recognition and video retrieval tasks. However, our pretext samples an unlabeled clip and selects a scene to be repeated without considering the high-motion regions in the video, which could enrich the visual

Fig. 8. RSL video retrieval. Our R(2+1)D focuses on the actions motion dynamics.

representations further. In future work, our proposed pretext could be improved by detecting the high-motion segments in the video and using them to sample a clip and generate repeated scenes. In addition, we would like to explore the possibility of using the learned visual representations in other video understanding tasks, such as action detection and action localization.

CRediT authorship contribution statement

Hussein Altabrawee: Conceptualization, Methodology, Writing – original draft. **Mohd Halim Mohd Noor:** Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The datasets are available publicly.

Acknowledgments

This work has been supported in part by the Ministry of Higher Education Malaysia for Fundamental Research Grant Scheme with Project Code: FRGS/1/2023/ICT02/USM/02/2.

References

[1] H.-Y. Lee, J.-B. Huang, M. Singh, M.-H. Yang, Unsupervised representation learning by sorting sequences, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 667–676.

[2] I. Misra, C.L. Zitnick, M. Hebert, Shuffle and learn: unsupervised learning using temporal order verification, in: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, the Netherlands, October 11–14, 2016, Proceedings, Part I 14, Springer, 2016, pp. 527–544.

[3] A. El-Nouby, S. Zhai, G.W. Taylor, J.M. Susskind, Skip-clip: Self-supervised spatiotemporal representation learning by future clip order ranking, 2019, arXiv preprint arXiv:1910.12770.

[4] Y. Zhang, Z. Chen, T. Xu, J. Zhao, S. Mi, X. Geng, M.-L. Zhang, Temporal segment dropout for human action video recognition, Pattern Recognit. 146 (2024) 109985.

[5] Y. An, Y. Yi, X. Han, L. Wu, C. Su, B. Liu, X. Xue, Y. Li, A hybrid attention-guided ConvNeXt-GRU network for action recognition, Eng. Appl. Artif. Intell. 133 (2024) 108243.

[6] S. Rastegar, H. Doughty, C.G. Snoek, Background no more: Action recognition across domains by causal interventions, Comput. Vis. Image Underst. 242 (2024) 103975.

[7] Y. Ma, R. Wang, M. Zong, W. Ji, Y. Wang, B. Ye, Convolutional transformer network for fine-grained action recognition, Neurocomputing 569 (2024) 127027.

[8] Y. Ma, R. Wang, Relative-position embedding based spatially and temporally decoupled Transformer for action recognition, Pattern Recognit. 145 (2024) 109905.

[9] W. Jo, G. Lim, G. Lee, H. Kim, B. Ko, Y. Choi, VVS: Video-to-video retrieval with irrelevant frame suppression, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 38, 2024, pp. 2679–2687, (3).

[10] O. Thawakar, M. Naseer, R.M. Anwer, S. Khan, M. Felsberg, M. Shah, F.S. Khan, Composed video retrieval via enriched context and discriminative embeddings, 2024, arXiv preprint arXiv:2403.16997.

[11] T. Tai, F. Zeng, Research on video retrieval technology based on multimodal fusion and attention mechanism, in: Proceedings of the 2023 7th International Conference on Electronic Information Technology and Computer Engineering, EITCE '23, Association for Computing Machinery, New York, NY, USA, 2024, pp. 470–474.

[12] G. Lv, Y. Sun, F. Nian, Video–text retrieval via multi-modal masked transformer and adaptive attribute-aware graph convolutional network, Multimedia Syst. 30 (1) (2024) 35.
[13] J. Guo, M. Wang, W. Wang, Y. Zhou, B. Song, DI-VTR: Dual inter-modal interaction model for video-text retrieval, J. Inf. Intell. (2024).
[14] I. Dave, R. Gupta, M.N. Rizve, M. Shah, Tclr: Temporal contrastive learning for video representation, Comput. Vis. Image Underst. 219 (2022) 103406.
[15] A. Diba, V. Sharma, R. Safdari, D. Lotfi, S. Sarfraz, R. Stiefelhagen, L. Van Gool, Vi2clr: Video and image for visual contrastive learning of representation, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 1502–1512.
[16] B. Fernando, H. Bilen, E. Gavves, S. Gould, Self-supervised video representation learning with odd-one-out networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 3636–3645.
[17] D. Xu, J. Xiao, Z. Zhao, J. Shao, D. Xie, 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. 10334–10343.
[18] T. Suzuki, T. Itazuri, K. Hara, H. Kataoka, Learning spatiotemporal 3d convolution with video order self-supervision, in: Proceedings of the European Conference on Computer Vision (ECCV) Workshops, 2018.
[19] D. Wei, J.J. Lim, A. Zisserman, W.T. Freeman, Learning and using the arrow of time, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 8052–8060.
[20] S. Jenni, G. Meishvili, P. Favaro, Video representation learning by recognizing temporal transformations, in: European Conference on Computer Vision, Springer, 2020, pp. 425–442.
[21] A. Jabri, A. Owens, A. Efros, Space-time correspondence as a contrastive random walk, Adv. Neural Inf. Process. Syst. 33 (2020) 19545–19560.
[22] L. Tao, X. Wang, T. Yamasaki, Self-supervised video representation learning using inter-intra contrastive framework, in: Proceedings of the 28th ACM International Conference on Multimedia, 2020, pp. 2193–2201.
[23] K. Soomro, A.R. Zamir, M. Shah, UCF101: A dataset of 101 human actions classes from videos in the wild, 2012, arXiv preprint arXiv:1212.0402.
[24] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, T. Serre, HMDB: a large video database for human motion recognition, in: 2011 International Conference on Computer Vision, IEEE, 2011, pp. 2556–2563.
[25] J. Wang, J. Jiao, Y.-H. Liu, Self-supervised video representation learning by pace prediction, in: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVII 16, Springer, 2020, pp. 504–521.
[26] T. Han, W. Xie, A. Zisserman, Video representation learning by dense predictive coding, in: Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops, 2019.
[27] D. Luo, C. Liu, Y. Zhou, D. Yang, C. Ma, Q. Ye, W. Wang, Video cloze procedure for self-supervised spatio-temporal learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, 2020, pp. 11701–11708.
[28] T. Han, W. Xie, A. Zisserman, Memory-augmented dense predictive coding for video representation learning, in: European Conference on Computer Vision, Springer, 2020, pp. 312–329.
[29] J. Wang, Y. Lin, A.J. Ma, P.C. Yuen, Self-supervised temporal discriminative learning for video representation learning, 2020, arXiv preprint arXiv:2008.02129.
[30] H. Cho, T. Kim, H.J. Chang, W. Hwang, Self-supervised spatio-temporal representation learning using variable playback speed prediction, 2020, pp. 13–14, arXiv preprint arXiv:2003.02692 2.
[31] Y. Yao, C. Liu, D. Luo, Y. Zhou, 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.
[32] W. Li, D. Luo, B. Fang, Y. Zhou, W. Wang, Video 3d sampling for self-supervised representation learning, 2021, arXiv preprint arXiv:2107.03578.

Hussein Altabrawee is a Ph.D. candidate in computer science at the Universiti Sains Malaysia. He has worked as a university lecturer in computer science at Al-Muthanna University from 2014 to 2020. Hussein has received a Bachelor degree in computer science from the University of Babylon in 2007. He has studied computer science at California State University Fullerton and received a Master of Science degree in 2013. His research interest focuses on machine learning, deep learning, and computer vision.

Mohd Halim Mohd Noor received the B.Eng. degree (Hons.) in 2004, the M.Sc. degree in 2009, and the Ph.D. degree in computer systems engineering from the University of Auckland, New Zealand, in 2017. He is currently a Senior Lecturer with the School of Computer Sciences, Universiti Sains Malaysia. His research interests include machine learning, deep learning, computer vision, and pervasive computing.