ASCNet: Self-supervised Video Representation Learning with Appearance-Speed Consistency

Deng Huang1,3*, Wenhao Wu2*, Weiwen Hu1, Xu Liu1, Dongliang He2, Zhihua Wu2, Xiangmiao Wu1, Mingkui Tan1,4†, Errui Ding2

1 South China University of Technology 2 Baidu Inc. 3 Pazhou Laboratory 4 Key Laboratory of Big Data and Intelligent Robot, Ministry of Education

{sehuangdeng,sehuww,seqdmy}@mail.scut.edu.cn, {xmwu, mingkuitan}@scut.edu.cn, wuwenhao17@mails.ucas.edu.cn, {hedongliang01, wuzhihua02, dingerruis}@baidu.com

Abstract

We study self-supervised video representation learning, which is a challenging task due to 1) lack of labels for explicit supervision; 2) unstructured and noisy visual information. Existing methods mainly use contrastive loss with video clips as the instances and learn visual representation by discriminating instances from each other, but they need a careful treatment of negative pairs by either relying on large batch sizes, memory banks, extra modalities or customized mining strategies, which inevitably includes noisy data. In this paper, we observe that the consistency between positive samples is the key to learn robust video representation. Specifically, we propose two tasks to learn appearance and speed consistency, respectively. The appearance consistency task aims to maximize the similarity between two clips of the same video with different playback speeds. The speed consistency task aims to maximize the similarity between two clips with the same playback speed but different appearance information. We show that optimizing the two tasks jointly consistently improves the performance on downstream tasks, e.g., action recognition and video retrieval. Remarkably, for action recognition on the UCF-101 dataset, we achieve 90.8% accuracy without using any extra modalities or negative pairs for unsupervised pretraining, which outperforms the ImageNet supervised pretrained model. Codes and models will be available.

1. Introduction

By 2022, almost 79% of the world’s mobile data traffic will be video. With the rise of cameras on smartphones, recording videos has never been easier. Video analysis has become one of the most active research topics [36, 35, 37]. However, the generation of high-quality video data requires a massive human annotation effort (e.g., Kinetics-400 [20], Youtube-8M [1]), which is expensive, time-consuming, and hard to scale up. In contrast, millions of unlabeled videos are freely available on the Internet, e.g., YouTube. Thus, learning meaningful representations from unlabeled videos is crucial for video analysis.

Self-supervised learning makes it possible to exploit a variety of labels that come with the data for free. Instead of collecting manual labels, the proper learning ob-
jectives are set to obtain supervision from the unlabeled data themselves. These objectives, also known as pretext tasks, roughly fall into three categories: 1) predicting specific transformations (e.g., rotation angle [19], playback speed [2], and order [39]) of video clips; 2) generative dense prediction, e.g., future frame prediction [13]; and 3) instance discrimination, e.g., CVRL [27] and Pace [34]. Among these methods, the playback speed prediction task has attracted more attention because 1) we can easily train a model with speed labels generated automatically from the video inputs, and 2) models will focus on the moving objects to perceive the playback speed [34]. Thus, models are encouraged to learn representative motion features.

While promising results have been achieved, existing methods suffer from two limitations. First, some of the approaches rely on pre-computed motion information (e.g., optical flow [15, 33]), which is computationally heavy, particularly when the dataset is scaled up. Second, while negative samples play important roles in instance discrimination tasks, it is hard to maintain their quality and quantity. Moreover, same-class negative samples can be harmful [4] to the representations used in downstream tasks.

In this work, we explore the consistency between both appearance and speed of video clips from the same and different instances and eliminate the need for negatives that are detrimental in some cases. To this end, we propose two new pretext tasks, namely, Appearance Consistency Perception (ACP) and Speed Consistency Perception (SCP). Specifically, for the ACP task, we sample two clips from the same video with independent data augmentations and encourage the representations of the two clips to be close enough in feature space. Models cannot finish this task by learning low-level information, i.e., color and rotation. Instead, models tend to learn appearance features such as background scenes and the texture of objects because these features are consistent along a video. For the SCP task, we sample two clips from two different videos with the same playback speed. Representations of these two clips are pulled closer in the feature space. Since the appearance varies from instance to instance, speed can be the crucial clue to finish this task.

Moreover, to enrich the positive samples and integrate the ACP and SCP tasks, we propose an appearance-based video retrieval strategy, which is based on the observation that appearance features in the ACP task achieve a decent accuracy (45% top-1) in the video retrieval task. Thus, we collect the video with the same speed and similar appearance for the SCP task and make it more compatible with the ACP task. This strategy further improves the performance of downstream tasks with negligible computational cost.

To summarize, our contributions are as follows:

• We propose the ACP and SCP tasks for unsupervised video representation learning. In this sense, negative samples no longer affect the quality of learned representations, making the training more robust.
• We propose the appearance-based feature retrieval strategy to select the more effective positive sample for speed consistency perception. In this way, we can bridge the gap between two pretext tasks.
• We verify the effectiveness of our method for learning meaningful video representations on two downstream tasks, namely, action recognition and retrieval, on the UCF-101 [28] and HMDB51 [22] datasets. In all cases, we demonstrate state-of-the-art performance over other self-supervised methods, while our method is easier to apply in practice because we do not have to maintain the collection of negative samples.

2. Related Work

Self-supervised image representation learning. Self-supervised visual representation has recently witnessed tremendous progress on images. A common workflow is to train an encoder on one or multiple pretext tasks with unlabeled images and then use one intermediate feature layer of this model to feed a linear classifier on image classification. The final classification accuracy qualifies how good the learned representation is. The pretext tasks include image rotations [11], jigsaw puzzle [26] and colorization [41]. Recent progress of self-supervised learning is mainly based on instance discrimination, which maintains relative consistency between the representations of the anchor image and its augmented view. The performance of contrastive learning relies on a rich set of negative samples [6, 8, 17]. SimCLR [6] uses a large batch size and picks negatives in a minibatch. MoCo [17] maintains a large dictionary that covers negative features as a memory bank. Nevertheless, same-class negatives are inevitable and harmful to the performance of contrastive learning [4]. Recently, beyond contrastive learning, BYOL [12] and SimSiam [7] learn meaningful representation by only maximizing the similarity between two augmented positive samples without collapsing.

Self-supervised video representation learning. In recent years, video analysis has become a popular topic. Unlike static images, videos offer additional opportunities for learning meaningful representation by exploiting spatiotemporal information. Existing methods learn representation through various carefully designed pretext tasks. Some pretext tasks are extended from image-based representation learning, e.g., rotation prediction [19], jigsaw [21]. Other approaches pay more attention to temporal information, including sorting video frames or clips [23]. BE [32] erases the influence of video background by mixing a static frame with the whole clip and removing it. More recently, several attempts have been made through playback speed prediction. SpeedNet [2] predicts whether the video clip is
we use a video encoder $f(\cdot; \theta)$ to map the clips into appearance and speed embedding space. For the ACP task, we pull the appearance features from the same video closer. For the SCP task, we first retrieve the same speed video with similar contents and then pull the speed features closer. All of the video encoders share the parameters.

3. Approach

Problem definition. We let $\mathcal{V} = \{v_i\}_{i=1}^N$ be a set of training videos, and we sample a clip $c_i$ from $v_i$ with playback speed $s_i$. Self-supervised video representation learning aims to learn an encoder $f(\cdot; \theta)$ to map the clip $c_i$ to consistent feature $x_i$ under different video augmentations.

This task is extremely difficult because of insufficient labels and complex spatiotemporal information. First, it is difficult to construct proper supervision from unlabeled videos for models to learn both appearance and motion information. Second, it is inefficient to capture motion information from video, e.g., by computing optical flow in a sequence of frames. Consequently, the learned representation may not fulfill the requirements of downstream tasks, such as action recognition and video retrieval.

3.1. General Scheme of ASCNet

In this paper, we observe that video playback speed not only is a good source of temporal data augmentation that does not change the spatial appearance but also provides effective supervision for video representation learning. Thus, we propose Appearance Consistency Perception (ACP) task for learning appearance features, i.e., predict consistent appearance features under different spatiotemporal augmentations of the same video, and the Speed Consistency Perception (SCP) task for learning speed features, i.e., predict consistent speed features for different videos with the same playback speed.

Formally, for the ACP task, different from training the model to predict whether or not two clips $c_i$ and $c_j$ are sampled from the same video, we propose to minimize the distance between the representations of clips $c_i$ and $c_j$ in the embedding space. Our intuition is that the clips sampled from the same video naturally share similar appearance contents. We also randomize the playback speed so that $s_i$ can be equal or not to $s_j$. In this way, models are encouraged to learn appearance consistency. For the SCP task, we enforce the models to encode the playback speed information and shorten the distance between the representations of clips $c_i$ and $c_j$, which are sampled from different videos with $s_i = s_j$. In this manner, models are encouraged to learn the information that they have in common, namely, the playback speed.

We use two individual projection heads $g_a$ and $g_m$ to map the representation from $f$ to task corresponding features $a_i, a_j, m_i, m_j$, where $a_i, a_j$ are the features for the ACP task and $m_i, m_j$ are the features for the SCP task. The overall objective function of our Appearance-Speed Consistency Network (ASCNet) is formulated as follows:

$$L(\mathcal{V}) = \gamma L_a(\mathcal{V}) + (1 - \gamma) L_m(\mathcal{V}),$$  \hspace{1cm} (1)

where $L_a$ and $L_m$ denote the loss function of the ACP and SCP tasks, respectively. $\gamma$ is a hyperparameter that controls the importance of each task. The pretrained encoder $f(\cdot; \theta)$ and its output feature $x$ will be used in downstream tasks.

3.2. Appearance Consistency Perception

This task aims to minimize the representation distance between two augmented clips from the same video. Given
a video, we sample two clips $c_i, c_j$ with different playback speeds $s_i, s_j$, respectively. We feed the clips into the video encoder $f$ followed by a projection head $g_a$ to obtain the corresponding features $a_i, a_j$. Following common practice [12, 7], we pass $a_i$ to an additional predictor $h_a$ to obtain the prediction $a'_i$. We also employ a momentum target encoder, whose parameters $\xi$ is an exponential moving average (EMA) of the corresponding parameters $\theta$. We minimize the feature distance by using $l_2$ loss as follows:

$$a'_i = h_a(g_a(f(c; \theta); \theta_a); \theta'_a)$$
$$a_j = g_a(f(c_j; \xi); \xi_a)$$
$$\mathcal{L}_a = ||a'_i - a_j||^2_2.$$ (2)

Since different data augmentations and playback speeds do not change the content of the clip, we expect the appearance features $a'_i$ and $a_j$ to be always similar.

### 3.3. Speed Consistency Perception

Temporal information is crucial for the downstream tasks, e.g., action recognition. Recently, video playback speed prediction has been used as a successful pretext task for perceiving temporal information [2, 34]. However, direct prediction of speed may be suboptimal for learning effective representation because the changes of some motion may be not obvious under different playback speeds. Thus, we propose the consistent speed perception task. This task aims to minimize the distance between two clips with the same playback speed while the appearance can be different. Specifically, we sample two clips $c_i, c_k$ from two videos $v_i, v_k$ in the video set $\mathcal{V}$ with $s_i = s_k$. Then, these two clips are processed similarly to the ACP task, except that we use projection head $g_m$ and predictor $h_m$ independent from those in the ACP task. Finally, we define the following $l_2$ loss between the prediction and its target feature:

$$m'_i = h_m(g_m(f(c_i; \theta); \theta_m); \theta'_m)$$
$$m_k = g_m(f(c_k; \xi); \xi_m)$$
$$\mathcal{L}_m = ||m'_i - m_k||^2_2.$$ (3)

However, for two dissimilar videos, the optimization of $\theta$ can be difficult, and it takes more time for the model to converge. Thus, we propose an appearance-based feature retrieval framework to collect similar videos in feature space.

#### 3.3.1 Appearance-based Feature Retrieval

The instance sampling strategy affects the performance of both the SCP and ACP tasks. The clips used in the SCP task can be sampled from the same instance or different instances. However, when using the former, the SCP task may fall back into the ACP task. Since some movements have their corresponding speed, i.e., running and jogging, the latter may lead to conflict between the ACP and SCP tasks. Thus, to reduce the conflict, we propose an appearance-based feature retrieval strategy as follows.

Given an anchor video $v_i$, we collect a candidate set of videos (gallery) $\mathcal{C} = \{v_1, v_2, ..., v_l\}$ from $\mathcal{V} \setminus v_i$. We sample a clip from each video to obtain the anchor feature $a_i$ and candidate features $\{a_1, a_2, ..., a_l\}$ by using the same process as the ACP task. A simple dot product function $d(., .)$ is used to measure the similarity between the anchor and candidates. Then, we sort the videos by their similarity score and select $v_k$ from the most similar candidates. In practice, we use the memory bank [17] to reduce the computation cost. The video pair $v_i, v_k$ can be used in the SCP task with the benefit of strong spatial augmentation while not breaking the appearance consistency.

Algorithm 1: Training method of ASCNet.

#### Require: Video set $\mathcal{V} = \{v_i\}_{i=1}^N$, the encoder $f$ with parameters $\theta$ or $\xi$, the projection heads $g_a$ and $g_m$ with parameters $\theta_a, \xi_a, \theta_m$ and $\xi_m$, the predictors $h_a$ and $h_m$ with parameters $\theta'_a$ and $\theta'_m$, the hyperparameter $\gamma$.

1. Randomly initialize parameters $\theta, \theta_m, \theta'_a, \theta'_m$.
2. Initialize parameters $\xi \leftarrow \theta, \xi_a \leftarrow \theta_a, \xi_m \leftarrow \theta_m$.
3. while not convergent do
   4. Randomly sample a video $v$ from $\mathcal{V}$.
   5. Sample two clips $c_i, c_j$ from $v$.
   6. Extract features $x_i = f(c_i, \theta), x_j = f(c_j, \xi)$.
   7. // Learn appearance features with ACP task
      Obtain $a_i = g_a(x_i, \theta_a), a_j = g_a(x_j, \xi_a)$.
      Obtain $a'_i = h_a(a_i, \theta'_a)$.
   8. Compute $\mathcal{L}_a = ||a'_i - a_j||^2_2$ in Eqn. (2).
   9. // Conduct appearance-based feature retrieval
      Construct $\mathcal{C} = \{a_j\}_{j=1}^{N-1}$ using $g_a(., \theta_a)$ from $\mathcal{V} \setminus v$.
      // Obtain $C$ efficiently from memory bank
   10. Select the video $\hat{v}$ corresponding to features $a \in \mathcal{A}$ with the highest dot product similarity with $a_j$.
   11. // Learn speed features with SCP task
      Sample one clip $c_k$ from $\hat{v}$.
      Extract features $x_k = f(c_k, \xi)$.
   12. Construct $\mathcal{C} = \{a_j\}_{j=1}^{N-1}$ using $g_a(., \theta_a)$ from $\mathcal{V} \setminus v$.
   13. Compute $\mathcal{L}_m = ||m'_i - m_k||^2_2$ in Eqn. (3).
   14. Update parameters $\theta, \theta_m, \theta'_a, \theta'_m$ via SGD.
   15. Compute exponential moving average $\xi, \xi_a, \xi_m$.
   22. end while

4. Experiments

#### 4.1. Datasets

We consider four video datasets, including Mini-Kinetics-200 [38], Kinetics-400 [20], UCF-101 [38], and HMDB-51 [22]. For the self-supervised pretraining, we use
the training split of the Kinetics-400 dataset by discarding all of the labels. The Kinetics-400 dataset contains 400 human action categories and provides 240k training video clips and 20k validation video clips. The Mini-Kinetics-200 dataset consists of 200 categories with the most training examples and is a subset of the Kinetics-400 dataset. Since the full Kinetics-400 is quite large, we use Mini-Kinetics-200 to reduce training costs in ablation experiments.

The learned network backbones are evaluated via two downstream tasks: action recognition and nearest neighbor retrieval. For the downstream tasks, UCF-101 [28] and HMDB-51 [22] are used to demonstrate the effectiveness of our method. UCF-101 [28] contains 13k videos spanning over 101 human actions. HMDB-51 [22] contains approximately 7k videos belonging to 51 action classes categories. Both UCF-101 [28] and HMDB-51 [22] come with three predefined training and testing splits. Following prior work [39, 2, 34], we use a training/testing split of 1 for downstream task evaluation. Both datasets exhibit challenges including intraclass variance of actions, cluttered backgrounds, and complex camera motions. Performing action recognition and retrieval on these datasets requires learning rich spatiotemporal representation.

4.2. Implementation Details

Backbone networks. To study the effectiveness and generalization ability of our method in detail, we choose three different backbone networks as the video encoder, which have been widely used in recent video self-supervised learning methods, i.e., 3D ResNet [16], R(2+1)D [31], and S3D-G [38]. 3D ResNet [16] is a natural extension of the ResNet architecture [18] for directly tackling 3D volumetric video data by extending 2D convolutional kernels to the 3D counterparts. R(2+1)D [31] is proposed to decompose the full 3D convolution into the 2D spatial convolution followed by the 1D temporal convolution. Moreover, following previous work [2, 34], we use the state-of-the-art backbone S3D-G [38] to further exploit the potential of the proposed approach.

Self-supervised pretraining stage. Following prior work [34, 2, 5], we sample 16 consecutive frames with 112 × 112 spatial size for each clip unless specified otherwise. Video clips are augmented using random cropping with resizing, random color jittering, random Gaussian blurring, and random grayscale and solarization. We utilize LARS as the optimizer with a momentum of 0.9 and weight decay of 1e-6 for training without dropout operation. We set the base learning rate to 0.3, scaled linearly with the batch size $b$; i.e., the learning rate is set to $0.3 \times b/128$. The pretraining process is carried out for 200 epochs by default. The possible playback speed $s$ for the clips in this paper is set to $\{1 \times, 2 \times, 4 \times, 8 \times\}$, i.e., sampling frames consecutively or setting the sampling interval to $\{2, 4, 8\}$ frames. We use only the raw unfiltered RGB video frames as the input and do not make use of optical flow or other auxiliary signals during training. Moreover, we instantiate all projection heads as a fully connected layer with 256 output dimensions. We apply L2 normalization for all features. After pretraining, we drop the projection heads and use the features for downstream tasks. When jointly optimizing $L_a$ and $L_m$, we empirically set the parameters $\gamma$ as 0.5 for loss balance.

Supervised fine-tuning stage. Regarding the action recognition task, during the fine-tuning stage, the learning rate was decayed by a factor of 0.01 over the course of training using cosine annealing. The weights of convolutional layers are retained from the learned representation model, while the weights of the newly appended fully connected layers are randomly initialized. The whole network is then trained with cross-entropy loss.

Evaluation. During inference, following the common evaluation protocol, we sample 10 clips uniformly from each video in the test sets of UCF-101 and HMDB-51. For each clip, we only simply apply the center-crop instead of the ten-crop. Finally, we average the softmax probabilities of all clips as the final prediction.

4.3. Ablation Studies

As shown in Table 1, we provide ablation studies on the effectiveness of the different aspects of our method by self-supervised learning on Mini-Kinetics-200. The representations are evaluated on UCF-101 with end-to-end fine-tuning. The analysis was carried out as follows.

Effectiveness of ASC. In this paper, we propose two tasks to learn effective video representation, namely, Appearance Consistency Perception (ACP) and Speed Consistency Perception (SCP). To verify the effectiveness of our method, we pretrain these models with 3D ResNet-18. As shown in Table 1a, compared with training from scratch, pretraining with only the ACP task can significantly improve the action recognition performance (64.76% vs. 42.40%) on the UCF-101 dataset, while consistent speed perception further improves the performance from 64.76% to 70.71%, indicating the effectiveness of cooperative work of these two tasks. In the following ablation experiments, unless stated otherwise, we apply 3D ResNet-18 (3D R18) as the backbone.

Ablation on SCP tasks. Here, we instantiate some variants of our method by using different speed perception tasks [2]. SP denotes speed prediction for each individual clip. Table 1b shows that the speed consistency perception task improves the performance compared with directly predicting the playback speed of each clip (70.71% vs. 68.93%). Then, we investigate the instance sampling strategy for the SCP task. The video clips used in the SCP task can be sampled from the same instance or different instances. Similar instances denotes using the appearance-
based feature retrieval strategy. These results demonstrate that the appearance-based feature retrieval strategy can beneﬁt the speed consistency perception task while not breaking the appearance consistency.

**Different playback speed.** We denote the $s_i, s_j$ in Algorithm 1 as $\{S_1, S_2\}$. As shown in Table 1c, we compare the performance of different playback speed sets $\{S_1, S_2\}$ for our method. In particular, when the speed set is $\{\times 1, \times 1\}$, our ASC loses the speed perception and degenerates to pay more attention to learning appearance information. As expected, the performance decreases from 70.71% of $\{\times 1, \times 2\}$ to 64.52%, which is similar to the 64.76% of ACP Only in Table 1a. Then, when the playback speed $S_i$ is set to $\times 1$, we observe that the changes in $S_2 = \times 2, \times 4$ appear to have little impact on performance. Interestingly, for $\{\times 4, \times 8\}$, larger sampling intervals encourage the model to explore longer motion information, boosting the learned representation (70.71% vs. 72.16%). Thus, we adopt it in the following experiments.

**Impact of batch size.** The ablation study of different batch sizes is shown in Table 1d. When the batch size changes, we use the same linear scaling rule for all batch sizes studied. Our method works reasonably well over the wide range of batch sizes without using negative pairs. Our experimental results show that a batch size of 256 already achieves high performance. The performance remains stable over a range of batch sizes from 256 to 1024, and the differences are at the level of random variations.

**Augmentation.** The accuracies for applying the following data augmentations during pretraining one-by-one are shown in Table 1e. With only color jittering, our ASC yields 62.43% accuracy. Then, we randomly blur the frames using a Gaussian distribution, boosting the accuracy by 2.1%. Random grayscale is the augmentation approach that converts the frames to grayscale with probability $p$ (default 0.2 in this paper). Equipped with random grayscale, the accuracy of ASC is improved from 64.52% to 67.22%. Finally, we solarize RGB/grayscale video frames by inverting all of the pixel values above a threshold, further improving the performance of our ASC to 72.16%. Overall, by stacking these augmentations, we have steadily improved the learned representation model from 62.42% to 72.16%. Thus, all these data transformations are used in our experiments.

**Different backbone.** Since it is a general framework, ASC can be widely applied to existing video backbones with consistent gains in performance. In Table 1f, we compare various instantiations of our framework and show that our method is simple yet effective. We observe a consistent improvement of between 20% and 30% on UCF-101 with our ASCNet on three video decoders, i.e., 3D ResNet-18 [16], R(2+1)D [31], and S3D-G [38].

**4.4. Evaluation on the Action Recognition Task**

**Different evaluation protocols.** We survey existing self-supervised video representation learning methods and make the following observations about the evaluation protocols: (1) Different works may use different cropping strategies for evaluation, such as center-crop [2, 34, 5], three-crop [27], and ten-crop [13, 14, 15]. (2) Even with the same backbone, many methods may use different resolutions (i.e., $112^2, 128^2, 224^2, 256^2$) or numbers of frames (i.e., 16, 32, 64) during evaluation. For readers’ reference, in Table 3, we present the results of our method with different evaluation protocols used in prior works.
Table 2: Comparison with state-of-the-art self-supervised learning methods on the UCF-101 and HMDB-51 datasets. The dataset parentheses show the total video duration (d for day, y for year). Single-Mod denotes the Single RGB Modality. K400 represents Kinetics-400.

Table 3: Performance of different evaluation protocols. The models are pretrained with 200 epochs on Kinetics-400.

Table 4: Performance of different pretraining epochs. A 3D ResNet-18 backbone with ASC pretraining is used.

Impact of the pretraining epochs. We experiment with pretraining epochs varying from 100 to 400 and report the top-1 accuracy on UCF-101. Table 4 shows the impact of the number of training epochs on performance. While ASC benefits from longer training, it already achieves strong performance after 200 epochs, i.e., 80.52%. We also notice that performance starts to saturate after 300 epochs.

Comparison with the state of the art. In Table 2, we perform a thorough comparison with the state-of-the-art self-supervised learning methods and report the top-1 accuracy on UCF-101 [28] and HMDB-51 [22]. We show the pretraining settings for all approaches, e.g., pretraining dataset, backbone, number of input frames, resolution, and whether or not only the RGB modality is used. Here, we mainly list the models using RGB as inputs for fair comparisons. Since prior works use different backbones for experiments, we provide results of our ASCNet trained with two common architecture, i.e., 3D ResNet-18 [16], S3D-G [38].

Our ASCNet achieves state-of-the-art results on both the UCF-101 and HMDB-51 datasets. Specifically, when pretrained with the 3D ResNet-18 backbone, our method outperforms 3D-RotNet [19], ST-Puzzle [21], and DPC [13] by a large margin (80.5% vs. 62.9%, 65.8%, and 68.2%, respectively, on UCF-101 and 52.3% vs. 33.7%, 33.7%, and 34.5%). When utilizing S3D-G as the backbone, our ASCNet achieves better accuracy than SpeedNet [2], Pace [34], and RSPNet [5] (90.8% vs. 81.1%, 87.1%, and 89.9%, respectively, on UCF-101 and 60.5% vs. 48.8%, 52.6%, and 59.9%) under the same settings. Remarkably, without the need of any annotation for pretraining, our ASCNet outperforms the ImageNet [10] supervised pretrained model over
<table>
<thead>
<tr>
<th>Method</th>
<th>Architecture</th>
<th>k = 1</th>
<th>k = 5</th>
<th>k = 10</th>
<th>k = 20</th>
<th>k = 50</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPN [23]</td>
<td>CaffeNet</td>
<td>19.9</td>
<td>28.7</td>
<td>34.0</td>
<td>40.6</td>
<td>51.6</td>
</tr>
<tr>
<td>Buchler et al. [3]</td>
<td>CaffeNet</td>
<td>25.7</td>
<td>36.2</td>
<td>42.2</td>
<td>49.2</td>
<td>59.5</td>
</tr>
<tr>
<td>ClipOrder [39]</td>
<td>3D R18</td>
<td>14.1</td>
<td>30.3</td>
<td>40.0</td>
<td>51.1</td>
<td>66.5</td>
</tr>
<tr>
<td>SpeedNet [2]</td>
<td>S3D-G</td>
<td>13.0</td>
<td>28.1</td>
<td>37.5</td>
<td>49.5</td>
<td>65.0</td>
</tr>
<tr>
<td>VCP [24]</td>
<td>3D R18</td>
<td>18.6</td>
<td>33.6</td>
<td>42.5</td>
<td>53.5</td>
<td>68.1</td>
</tr>
<tr>
<td></td>
<td>R(2+1)D</td>
<td>19.9</td>
<td>33.7</td>
<td>42.0</td>
<td>50.5</td>
<td>64.4</td>
</tr>
<tr>
<td>Pace [34]</td>
<td>3D R18</td>
<td>23.8</td>
<td>38.1</td>
<td>46.4</td>
<td>56.6</td>
<td>69.8</td>
</tr>
<tr>
<td></td>
<td>C3D</td>
<td>31.9</td>
<td>49.7</td>
<td>59.2</td>
<td>68.9</td>
<td>80.2</td>
</tr>
<tr>
<td>RSPNet [5]</td>
<td>C3D</td>
<td>36.0</td>
<td>56.7</td>
<td>66.5</td>
<td>76.3</td>
<td>87.7</td>
</tr>
<tr>
<td></td>
<td>3D R18</td>
<td>41.1</td>
<td>59.4</td>
<td>68.4</td>
<td>77.8</td>
<td>88.7</td>
</tr>
<tr>
<td>Ours</td>
<td>3D R18</td>
<td><strong>58.9</strong></td>
<td><strong>76.3</strong></td>
<td><strong>82.2</strong></td>
<td><strong>87.5</strong></td>
<td><strong>93.4</strong></td>
</tr>
</tbody>
</table>

Table 5: Comparison with state-of-the-art methods for nearest neighbor retrieval task on the UCF-101 dataset as measured by the top-\(k\) retrieval accuracy (%).

These results indicate that the proposed pretext tasks help us to learn more discriminative features.

**Qualitative results for video retrieval.** We further provide some retrieval results as a qualitative study. In Figure 3, the top is the query video from the UCF-101 testing set, and the bottom shows the top-3 nearest neighbors from the UCF-101 training set. We successfully retrieve highly relevant videos with similar appearance and motion. This result implies that our method is able to learn both meaningful appearance and motion features for videos.

5. Conclusion

This work presents an unsupervised video representation learning framework named ASCNet, which leverages the appearance consistency throughout the frames of the same video and speed consistency between videos with the same fps. We train the model to map these clips to appearance and speed embedding space while maintaining consistency. We also propose an appearance-based retrieval strategy to reduce the conflict between the appearance and speed consistency perception tasks. Extensive experiments show that the features learned by ASCNet perform better on action recognition and video retrieval tasks. In the future, we plan to incorporate additional modalities into our framework.

Acknowledgments. This work was partially supported by National Natural Science Foundation of China (NSFC) 62072190, Ministry of Science and Technology Foundation Project (2020AAA0106901), Program for Guangdong Introducing Innovative and Entrepreneurial Teams 2017ZT07X183, CCF-Baidu Open Fund (CCF-BAIDU OF2020022).
References


[37] Wenhao Wu, Yuxiang Zhao, Yanwu Xu, Xiao Tan, Dongliang He, Zikang Zou, Jin Ye, Yingying Li, Mingde Yao, Zichao Dong, and Yifeng Shi. Dsanet: Dynamic segment aggregation network for video-level representation learning. In ACMMM, 2021. 1

[38] Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In ECCV, pages 318–335, 2018. 4, 5, 6, 7

