Auxiliary Learning for Self-Supervised Video Representation via Similarity-based Knowledge Distillation

Amirhossein Dadashzadeh
University of Bristol
a.dadashzadeh@bristol.ac.uk

Alan Whone
University of Bristol
alan.whone@bristol.ac.uk

Majid Mirmehdi
University of Bristol
majid@cs.bris.ac.uk

Abstract

Despite the outstanding success of self-supervised pretraining methods for video representation learning, they generalise poorly when the unlabeled dataset for pretraining is small or the domain difference between unlabelled data in source task (pretraining) and labeled data in target task (finetuning) is significant. To mitigate these issues, we propose a novel approach to complement self-supervised pretraining via an auxiliary pretraining phase, based on knowledge similarity distillation, auxSKD, for better generalisation with a significantly smaller amount of video data, e.g. Kinetics-100 rather than Kinetics-400. Our method deploys a teacher network that iteratively distils its knowledge to the student model by capturing the similarity information between segments of unlabelled video data. The student model meanwhile solves a pretext task by exploiting this prior knowledge. We also introduce a novel pretext task, Video Segment Pace Prediction or VSPP, which requires our model to predict the playback speed of a randomly selected segment of the input video to provide more reliable self-supervised representations. Our experimental results show superior results to the state of the art on both UCF101 and HMDB51 datasets when pretraining on K100 in apple-to-apple comparisons. Additionally, we show that our auxiliary pretraining, auxSKD, when added as an extra pretraining phase to recent state of the art self-supervised methods (i.e. VCOP, VideoPace, and RSPNet), improves their results on UCF101 and HMDB51. Our code is available at https://github.com/Plrbear/auxSKD.

1. Introduction

Self-supervised learning (SSL) methods have taken great strides recently in acquiring high-level semantic visual representations from unlabelled data, eliminating the cost of annotating large-scale datasets [5, 8, 11, 17, 42]. The most common approach in SSL is to use large-scale unlabelled data, such as Kinetics-400 [16], for pretraining network parameters, followed by finetuning on a downstream task with a limited amount of labelled data [5, 8, 9, 11, 42]. Despite the reduction in manual labelling, the performance of SSL methods is leveraged on huge unlabelled datasets, which demands high computational and memory costs, especially for large-scale video datasets. For example, it takes around two weeks to train MoCo [11] on Kinetics-400 for 300 epochs with two Nvidia RTX 2080TI GPUs. In fact, such computational costs place many state of the art (SotA) SSL approaches only in the realms of huge corporations who have such powerful resources [5, 9, 28], and this further becomes a subject of ethical fairness as well as carbon emission foot-

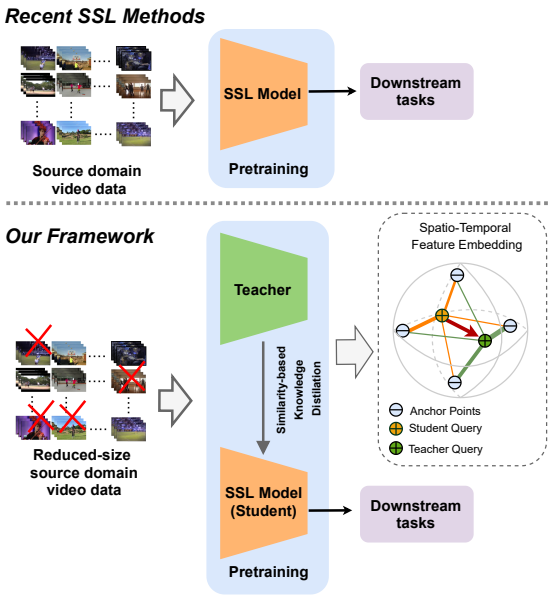

Figure 1. High-level overview of of our framework and recent SSL methods – while recent methods encourage their model to solve a pretext task from scratch, our SSL model benefits from an implicit similarity-based knowledge, distilled by a teacher model, before solving the pretext task. However, the question that we pose in this paper is: can we use an implicit knowledge of this type to improve the generalization ability of self-supervised approaches?

prints [34].

A few recent works have addressed the issue of efficient pretraining for *image-based* tasks [12, 22, 31], but there is only Lin et al. [21]'s work for *video-based* tasks which improves the generalization performance of a contrastive learning-based method [37] under a meta-learning paradigm. However, their method is not suited to most of the SotA works that use transformation-based pretext tasks [8, 15, 24, 41, 42].

Our motivation is to develop a *task-agnostic pretraining process that alleviates the dependency on large-scale datasets* for self-supervised video representation learning, while ensuring the model generalises well and still contains rich information. To achieve this, we propose an auxiliary pretraining stage, based on knowledge distillation (KD), which trains on a reduced version of the source dataset, provides implicit knowledge for the primary pretraining stage with the same reduced-size source dataset, and boosts generalization for downstream video representation learning tasks.

Figure 1 illustrates the difference between our framework and existing SSL methods, such as [5, 8, 15, 24, 42]. We employ a slowly progressing teacher model to iteratively distill knowledge to the student, our self-supervised model, by evaluating the similarity information of an augmented view of a query video clip to a large queue of random clips as anchors and transferring that information to the student. To the best of our knowledge, this is the first time such Similarity-based Knowledge Distilation (SKD) has been used in video-based self-supervised learning, while recently, SKD was adopted in image-based applications, for example for contrastive learning [35, 36] or model compression [1, 7]. To refer to this aspect of our work, we use auxSKD. To support the operation of the proposed approach on temporal features in the video domain, we apply temporal augmentations, in addition to spatial augmentations, to generate different transformed versions of a query video. Such temporal transformations are the same as the pretext transformations used in the primary pretraining stage. Their application at this stage allows our teacher to impart knowledge which matters most in the primary pretraining stage.

Also in this paper, we propose a new pretext task for video representation learning, namely Video Segment Pace Prediction (VSPP). While recent video playback rate prediction methods randomly sample training clips at different paces or speeds [3, 42], we sample training clips where only a randomly selected segment of the video has a randomly selected speed and the other segments of the video retain their natural pace. VSPP then requires the learner model to predict the playback speed of this randomly selected segment and its temporal location in the input training video. We advocate that by solving this pretext task, our model can strengthen its awareness of the natural pace of the clip and deal with the imprecise video speed labeling problem [5].

In our experiments, we show that our results, *based on Kinetics-100 pretraining as an example of a reduced-size dataset,* rather than the commonly used Kinectics-400, beat VCOP [43] (↑4.7% on UCF101 and ↑7.8% on HMDB51), VideoPace [42] (↑5.4% on UCF101 and ↑9.5% on HMDB51) and RSPNet [5] (↑2.7% on UCF101 and ↑2.3% on HMDB51) in like-for-like comparisons.

Our key contributions can be summarised as follows:

(i) we propose an auxiliary pretraining stage for self-supervised video learning to alleviate the dependency on large-scale source datasets, e.g. to allow using Kinetics-100 instead of Kinetics-400, (ii) we extend similarity-based knowledge distillation to the task of self-supervised video representation learning which has not been shown before, (iii) we show that our approach can benefit other pretext tasks for self-supervised pretraining that involve video transformations (e.g. VCOP [43], VideoPace [42] and RSPNet [5]), (iv) we propose a simple, yet effective and novel pretext task which is more commensurate with video motion events than existing video playback prediction tasks, (v) we achieve SotA results *when pretraining with Kinetics-100,* and evaluate the performance of different components of our proposed method through ablation experiments.

2. Related Work

In this section, we pay attention to three of the most prominent areas within the main scope of our work, with focus on the more recent, SotA approaches.

Auxiliary Learning – To assist a primary task to generalise better to unseen data, training through auxiliary learning is an effective approach [22, 26, 32] and has been applied alongside a wide range of techniques, such as transfer learning [40], reinforcement learning [20], semi-supervised learning [49], and knowledge distillation [44]. In a recent, work, Xu et. al [43] treat a self-supervised learning method as an auxiliary task for knowledge distillation, pushing the student to mimic the teacher on both classification output and auxiliary self-supervision pretext task output. They show that the auxiliary learning step regularizes the student to generalize better on few-shot and noisy-label scenarios. Here, we show that the similarity information between embedded feature points can be used as implicit knowledge for self-supervised pretraining to learn more generalised representations through pretext tasks. To capture this similarity information, we employ a variation of knowledge distillation, called similarity-based knowledge distillation [36, 39], as an auxiliary task which can be applied to any self-supervised pretraining method.

Similarity-based Knowledge Distilation – Knowledge distillation [4, 13] establishes a framework for improving the performance of a lightweight student model, guided by a larger and better performing teacher model which distills

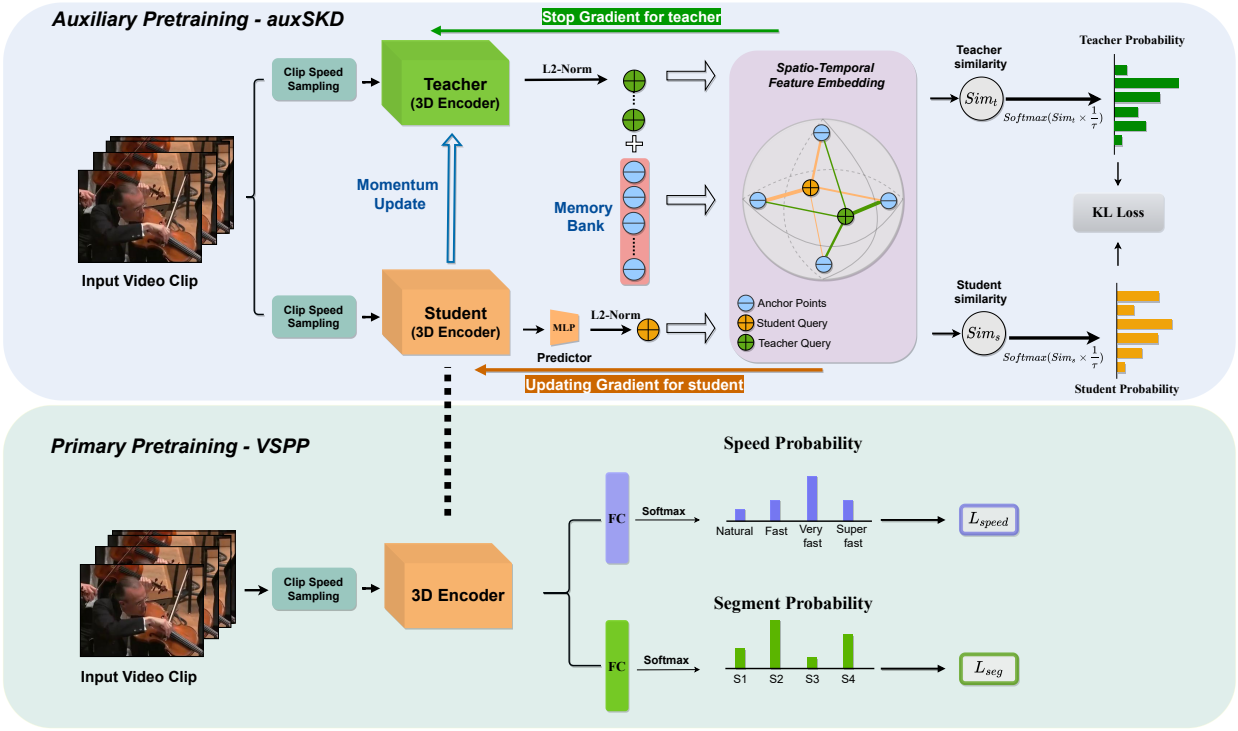

Figure 2. The self-supervised learning pretext training scheme is supported by an Auxiliary Pretraining task (auxSKD - see top region) that provides a similarity knowledge distillation process via a teacher-student configuration. In this configuration, both the teacher and the student 3D encoders are initialized and trained from scratch. Our teacher encoder is updated using momentum as a moving-average of the student weights. We train the student via gradient update by minimizing the KL divergence between the two probabilities from the teacher and the student for a transformed version of input video v, computing its similarity over anchor points. Note that in each iteration our encoders randomly take a different transformed input via our clip speed sampling process (see section 3.2). In the primary pretraining task (see bottom region), the student is ready to solve our VSPP task on input clips with segments that include changed pace.

the (dark) knowledge it learns to its student [13, 27, 47, 48].

Similarity-based Knowledge Distillation (SKD) methods [1, 7, 29, 30, 35, 36, 39] train a student to mimic the similarity score distribution inferred by the teacher over data samples. Most early works in SKD use a supervised loss during distillation [29, 30, 39], but there are a number of works that combine SKD with contrastive SSL learning to achieve SSL model compression and performance gains [1, 7]. While these works rely on a pre-trained frozen teacher model, Tejankar et al. [35] propose an iterative SKD regime where the teacher model continues to learn similarity score distributions during training. Our auxSKD architecture is similar to [35], but our objectives are different, (i) we expand SKD to extract representations from video data, instead of images, and (ii) we use distilled similarity representations as auxiliary knowledge for different self-supervised pretraining methods, instead of directly applying them on downstream tasks.

Video Playback Speed Perception – Creating an efficient self-supervised pretext task to model motion and appearance for video SSL is significantly more difficult than for static images [5]. Recently, estimating video playback speed has attracted much interest as a highly effective way to encourage the model to learn features (of moving objects) in videos [3, 5, 6, 14, 15, 42, 46]. For example, Epstein et al. [6] design a method to predict normal video speed to detect an unintentional event in the video. SpeedNet [3] determines whether a given video clip is being played at normal or twice its original speed, while VideoPace [42] predicts the specific speed of each video clip which is randomly sampled at a different frame rate. One of the main limitations in considering playback speed alone is that video clips with different speed labels might appear similar to each other, e.g. when different athletes might perform the same sporting action at different speeds.

To avoid the dependence on imprecise speed labels, Chen et al. [5] introduce RSPNet to predict relative speed between two video clips to better learn motion features. They use a triplet loss to minimize the distance between two clips of the same video at the same playback speed and maximize the distance between two clips of the same video at different playback speeds. Also in ASCNet [14], Hunag

et al. focus on speed similarity and propose a consistent speed perception task which aims to minimize the distance between two clips from two different videos with the same playback speed.

In VSPP, we propose a simple, yet effective video sampling strategy which does not rely on comparing video clips at different speed rates, since we embed a speed rate change within each clip. This emphasises focus on motion and abstracts the model from appearance features for which RSPNet and ASCNet require a whole additional processing pipeline. It also allows our model to converge much faster, e.g. after 20 epochs compared to 200 for RSPNet and ASCNet.

3. Proposed Approach

Our goal is to reduce the pretext training computational burden by developing an auxiliary pretraining phase that assists the primary pretext task to learn as efficient generalised self-supervised video representation as possible on a reduced-size source dataset. To achieve this, we take inspiration from Similarity-based Knowledge Distillation which is used in recent works [1, 7, 35, 36]. We illustrate our full self-supervised pretraining framework in Figure 2.

3.1. Auxiliary Learning via auxSKD

Our auxiliary learning framework consists of a teacher $\mathcal{T}$ and a student $\mathcal{S}$ with the same architecture followed by a fully-connected layer, as the projection, to map the representations into a lower dimension space. We follow BYOL [9] and use a MLP predictor layer on top of the student model. This makes the teacher-student architecture asymmetric to prevent collapsed solutions. We randomly initialize both models from scratch equally. The student model and its predictor layer are updated by back-propagation while momentum update [11] is applied in the teacher model to be a running average of the student.

At each iteration our pretext task transformation VSPP is applied twice to a raw video instance v to generate two video clips v_1^* and v_2^* independently, with the goal of maximizing their similarity in our teacher-student framework. Then, given feature encodings $(\mathcal{T}(v_1^*), \mathcal{S}(v_2^*))$ and predictor function MLP(.) for $\mathcal{S}$, we perform L_2 normalization such that $z^{\mathcal{T}} = \mathcal{T}(v_1^*)/\|\mathcal{T}(v_1^*)\|_2$ and $z^{\mathcal{S}} = \text{MLP}(\mathcal{S}(v_2^*))/\|\text{MLP}(S(v_2^*))\|_2$.

Similar to [11,35], we consider a memory bank of H feature vectors (or anchors) $x_i^{\mathcal{T}} = [x_1^{\mathcal{T}}, ..., x_H^{\mathcal{T}}]$ obtained from the teacher model under a simple FIFO strategy. Specifically, at each iteration, we enqueue the feature vectors of the current batch extracted from the teacher model and dequeue the earliest instances. Next, we calculate the similarity of the teacher's embedding $z^{\mathcal{T}}$ to all feature vectors in the memory bank and apply Softmax to obtain a probability distribution,

$$p_i^{\mathcal{T}} = -\log \frac{\exp(\text{sim}(z^{\mathcal{T}}, x_i^{\mathcal{T}})/\gamma^{\mathcal{T}})}{\sum_{j=1}^{H} = \exp(\text{sim}(z^{\mathcal{T}}, x_j^{\mathcal{T}})/\gamma^{\mathcal{T}})}\,, \quad (1)$$

where $p_i^{\mathcal{T}} = [p_1^{\mathcal{T}}, ..., p_H^{\mathcal{T}}]$ is the probability of teacher query $z^{\mathcal{T}}$ for the i-th anchor point, sim(.,.) measures the similarity between L_2 vectors, and $\gamma^{\mathcal{T}}$ is the temperature value for the teacher's model.

Similarly, we calculate the student similarity distribution $p_i^{\mathcal{S}} = [p_1^S, ..., p_H^S]$ over anchor points, with

$$p_i^{\mathcal{S}} = -\log \frac{\exp(\text{sim}(z^{\mathcal{S}}, x_i^{\mathcal{T}})/\gamma^{\mathcal{S}})}{\sum_{j=1}^{H} = \exp(\text{sim}(z^{\mathcal{S}}, x_j^{\mathcal{T}})/\gamma^{\mathcal{S}})}\,. \quad (2)$$

Here $\gamma^{\mathcal{S}}$ is the temperature value for the student's model. Finally, the loss is measured by the Kullback–Leibler (KL) divergence as

$$\mathcal{L}(\mathcal{T}, \mathcal{S}) = \sum_i \text{KL}(p_i^{\mathcal{T}} \parallel p_i^{\mathcal{S}})\,. \quad (3)$$

Note that during training, the teacher network's weights are initialised randomly and then they evolve gradually as a running average of the student using momentum with the update rule $\theta_{\mathcal{T}} \leftarrow m\theta_{\mathcal{T}} + (1-m)\theta_{\mathcal{S}}$, where $m \in [0, 1)$ is the momentum hyperparameter to ensure smoothness and stability, and $\theta_{\mathcal{T}}$ and $\theta_{\mathcal{S}}$ are the teacher and student model parameters respectively. Pseudo-code for our auxSKD training is provided in Algorithm 1.

3.2. Primary Pretext task learning via VSPP

A SSL pretext task encourages the neural network to learn a representation from unlabelled data which contains high-level abstractions or semantics. In the video pace prediction approach of Wang et al. [42], each training clip is randomly sampled at a different pace and their pretext task then identifies the pace for each clip. While this is an effective approach, it means each clip is treated as if its pace is its natural speed.

We propose that each clip should contain within it one segment where the pace has been (randomly) altered. Our assumption is similar to [3, 5, 42] in that the network can only represent the underlying video content through efficient spatiotemporal features if it succeeds in learning the pace reasoning task, however, we build on [42]'s proposal through a more intricate, yet simple, within-video pace alteration task. Our proposed VSPP pretext task requires our model to temporally explore a video clip and predict the index and speed of a segment within a clip which is sampled at a different speed rate.

Given a video clip v_i comprising N frames, we generate video $v_i^* = \{I_0, I_1, ..., I_{K-1}\}$ of size $K < N$, comprising

Algorithm 1 Training auxSKD

Input: Teacher model $\mathcal{T}(.,\theta_{\mathcal{T}})$ and student model $\mathcal{S}(.,\theta_{\mathcal{S}})$, videos $V = \{v_i\}_{i=1}^{N}$, and memory bank $\{x_i^{\mathcal{T}}\}_{i=1}^{H}$.
Output: Trained student model weights $\theta_{\mathcal{S}}$.

Randomly initialize $\mathcal{T}(.,\theta_{\mathcal{T}})$ and $\mathcal{S}(.,\theta_{\mathcal{S}})$.
while not max epoch **do**
  Randomly sample a video v from V.
  Sample two clips v_1^* and v_2^* from v using VSPP video transformation (Section 3.2).
  Compute student query features $z^{\mathcal{S}}$ from clip v_1^* using student model $\mathcal{S}(.,\theta_{\mathcal{S}})$.
  Update teacher parameters using momentum: $\theta_{\mathcal{T}} \leftarrow m\theta_{\mathcal{T}} + (1-m)\theta_{\mathcal{S}}$.
  Compute teacher query features $z^{\mathcal{T}}$ from clip v_2^* using teacher model $\mathcal{T}(.,\theta_{\mathcal{T}})$.
  Calculate $p_i^{\mathcal{T}}$ and $p_i^{\mathcal{S}}$ using Eq. 1 and Eq. 2.
  Add the teacher's embedding $z^{\mathcal{T}}$ into the memory bank $\{x_i^{\mathcal{T}}\}_{i=1}^{H}$ and pop-out the earliest sample.
  Optimize student model using KL divergence loss, Eq. 3.
end while

Z segments, such that K/Z number of frames in segment ζ are sampled at pace λ, where both ζ and λ are randomly selected from $1 \leq \zeta \leq Z$ and $1 \leq \lambda \leq Q$ respectively, and Q is the highest possible speed rate. Note, when $Z = 1$ the sampling strategy is similar to [42]. In this work, we select $Z = 4$ and $Q = 4$ to allow a significantly wide variation of starting locations and sudden speed rate changes to provide more precise self-supervision signals. Specifically, our approach results in a change of speed rate in only one segment of the clip while the rest of the clip (before and after) retains its natural rate (see Figure 3). This strategy allows the network to better find the difference between natural speed (changes which happen gradually) and altered speed (changes which happen suddenly) in a clip, alleviating imprecise video speed labeling issues [5].

To have a random speed rate λ for the ζ^{th} segment, beginning at frame I_b and ending at frame I_e, then

$$I_b = f_r + ((\zeta - 1) * \frac{K}{Z}) + (\lambda - 1),$$
$$I_e = I_b + \lambda * (\frac{K}{Z} - 1)\,, \tag{4}$$

where f_r is the r^{th} frame of the original video v_i which is randomly selected during sampling to generate a more diverse video clip v_i^* at each iteration.

Summary of our overall method: Given a 3D encoder, such as an R(2+1)D or R3D-18, and a video dataset $V = \{v_i\}_{i=1}^{N}$, we perform our auxiliary learning stage using our flavour of SKD (see Section 3.1 and Algorithm 1) based on a KL loss (Eq. 3). Following this auxiliary pretraining stage, the student model enters the primary pretraining stage to solve our VSPP pretext task through two simultaneous sub-tasks: i) predicting the speed rate λ in the ζ^{th} segment of v^*, ii) predicting the temporal location of the segment in v^* which is sampled at a different speed, i.e. predicting index ζ. Then by jointly optimizing these two tasks, the final self-supervised loss is defined as:

$$\mathcal{L} = \alpha\mathcal{L}_{speed} + \beta\mathcal{L}_{seg}, \tag{5}$$

where α and β are balancing weights (empirically found to work best in our experiments when $\alpha = \beta = 1$). $\mathcal{L}_{speed}$ and $\mathcal{L}_{seg}$ are cross-entropy losses.

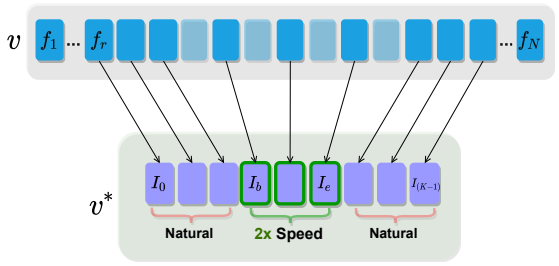

Figure 3. Changing the natural speed of one random segment of a video clip for the pretraining stage - the VSPP pretext task learns where in v^* this occurs and at what speed change. In this example $\lambda = 2$ and $\zeta = 2$.

4. Experiments

4.1. Datasets

We conducted our experiments on four datasets, two for pretraining, Kinetics-400 [16] (K-400) and Kinetics-100 [5] (K-100), and two for downstream action recognition, UCF101 [33], HMDB51 [18].

Kinetics-400 is a huge dataset for action recognition collected from Youtube, which contains 400 human action classes and around 240K videos. Each video lasts about 10 seconds.

Kinetics-100 comprises 100 classes and around 33K videos. We use K-400 and K-100 as pretraining datasets to validate our proposed approach's performance on a reduced-size dataset and promote less dependency on large-scale datasets for self-supervised representation learning.

UCF101 contains 101 action categories with a total of 13320 videos. It is divided into three training/testing splits and we follow prior works [5, 42] to use training split 1 for self-supervised pre-training and train/test split 1 for fine-tuning and evaluation.

HMDB51 contains 6,849 video clips of 51 action classes – a relatively small dataset compared to UCF101 and Kinetics. It is divided into three splits and we use split 1 for our downstream task evaluation, similar to [5, 42].

4.2. Implementation Details

Backbone Networks – We choose two different backbones R(2+1)D [38] and R3D-18 [10], as our 3D encoder, which have been widely used in recent SotA self-supervised video representation learning methods [5, 14, 28].

Default Settings – We run all experiments under PyTorch on two GeForce RTX 2080Ti GPUs with a batch size of 30. We use SGD as our optimizer with momentum of 0.9 and weight decay of 5e-4.

Pretraining Stages – Following [42], for both auxiliary and primary stages, we pretrain our models for 20 epochs with an initial learning rate of 1×10^{-3}. The learning rate is decreased by 1/10 every 6 epochs. For data augmentation, we randomly crop the video clip to 112 × 112 and then apply horizontal flip and color jittering to each video frame. Following [2], for UCF101 we apply (10x more iterations at) 90K iterations per epoch for temporal jittering. In our auxiliary pretraining stage, we use a predictor head for the student encoder comprising a 3-layer MLP with hidden dimension 1024, and output embedding dimension 128. We do not use a predictor for the teacher and only set its output dimension (projection head) to 128. We follow MoCo [28] and set the size of the memory bank to 16384 and set the momentum value of the encoder update to 0.999. We also use the same temperature for both teacher and student model at 0.02. To use the student encoder for the primary stage, the weights of the convolutional layers are retained after auxiliary pretraining and we drop the projection layer and predictor to replace them with two randomly initialized FC layers corresponding to the segment speed and index outcomes of our VSPP pretext task (see Figure 2). We select our parameters empirically.

Fine-tuning – During fine-tuning, we transfer the weights of the convolutional layers to the human action recognition downstream task, while the last FC layer is randomly initialized. We fine-tune the network on UCF101 and HMDB51 for 25 epoches with labelled videos and apply cross-entropy loss. We use the same data augmentation and training strategy as the pretraining stage except for the initial learning rate which is set to 3×10^{-3}, similar to [42].

Evaluation Settings – We follow the common evaluation protocols on video representation learning [14, 42] to assess the performance of our proposed approach. For action recognition, we sample 10 clips uniformly from each video in the test sets of UCF-101 and HMDB-51. Then for each clip, we only simply apply the center-crop. To find the final prediction, we average the Softmax probabilities of all 10 clips from the video.

4.3. Evaluations on UCF101 and HMDB51

Comparison on K400 pretraining – For completeness sake, and to illustrate how our proposed method fares when pretrained on K400, we present comparative results in Table 1 for top-1 accuracy on both UCF-101 and HMDB-51 datasets, along with the pretraining settings for all methods, i.e. backbone architecture, input size, pretraining dataset, and number of epochs. In Rows 1-4, we show a mix of methods that operate on temporal manipulations at different input sizes and on different backbones for reference. Rows 5-10 allow more like-for-like comparisons of recent, popular works in SSL video representation learning based on K-400 pretraining, R3D-18 backbone and almost consistent image sizes across the techniques. ASCNet achieves the most superior results with a combined appearance and speed manipulation approach. In Rows 11-15, where pretraining is on K-400 on the R(2+1)D architecture, RSPNet and VideoMoCo come 1st and 2nd-best alternatively on the two test datasets, while our approach exceeds CEP on both.

Comparison on K100 pretraining – The results on Rows 16-29 of Table 1 represent the essence of our contributions, in that we aim to reduce the dependence of SSL methods on large pretraining datasets, for example by replacing K-400 with K-100 for pretraining. We apply our auxSKD stage to two other transformation-based pretext tasks, i.e. VCOP, VideoPace, and also to one contrastive task, i.e. RSPNet, to exhibit the flexibility of our method. For VCOP and VideoPace, we train their auxSKD with video clips sampled based on VCOP and VideoPace's own sampling strategies, as proposed in [42] and [43] respectively. To integrate auxSKD into the RSPNet framework, we train it with the video transformation proposed in [5] and then transfer all the convolutional layer weights to its query encoder and initialize the projection head and key encoder randomly from scratch.

Rows 16-22 relate to the networks with a R(2+1)D backbone. VCOP+auxSKD improves on VCOP by ↑1.2% and ↑0.4% on UCF101 and HMDB51 respectively, while VideoPace+auxSKD similarly surpasses VideoPace alone by ↑2.3% and ↑2.4%. Note these are very close performances to when VideoPace is pretrained on K-400 (cf. Row 11). RSPNet's performance also improves when our auxiliary SKD is deployed, by ↑0.8% for UCF101 and ↑1.7% for HMDB51.

When the R3D-18 backbone is used, consistent improvements are again observed (see Rows 23-29) for all these methods when auxSKD is added to them. Our proposed method obtains the best performance using the R(2+1)D backbone on both datasets at 76.3% and 39.6%. When using R3D-18, it achieves the best result on UCF101 at 62.9% and the 2nd best on HMDB51 at 33.0%. Finally, we note that unlike VideoPace [42], RSPNet [5] and ASCNet [14] (for which no code has been released at the time of writing), we do not have an appearance stream in our method.

Row	Method	Network	Input Size	Pretrain	#epochs	Top 1 accuracy UCF101	HMDB51
1	Shuffle&Learn [25] [ECCV, 2016]	Alexnet	256 × 256	UCF101	-	50.2	18.1
2	OPN [19] [ICCV, 2017]	VGG	80 × 80	UCF101	-	59.8	23.8
3	VCOP [43] [CVPR, 2019]	C3D	112 × 112	UCF101	-	65.6	28.4
4	SpeedNet [3] [CVPR, 2020]	I3D	224 × 224	K-400	n/a	66.7	43.7
5	VideoPace [42] [ECCV, 2020]	R3D-18	112 × 112	K-400	18	63.7	27.9
6	VideoMoCo [28] [CVPR, 2021]	R3D-18	112 × 112	K-400	200	74.1	43.6
7	RSPNet [5] [AAAI, 2021]	R3D-18	112 × 112	K-400	200	74.3	41.8
8	ASCNet [14] [ICCV, 2021]	R3D-18	112 × 112	K-400	200	**80.5**	**52.3**
9	CEP [45] [BMVC, 2021]	R3D-18	224 × 224	K-400	50	75.9	36.6
10	Ours	R3D-18	112 × 112	K-400	40	67.9	32.6
11	VideoPace [42] [ECCV, 2020]	R(2+1)D	112 × 112	K-400	18	77.1	36.6
12	VideoMoCo [28] [CVPR, 2021]	R(2+1)D	112× 112	K-400	200	78.7	**49.2**
13	RSPNet [5] [AAAI, 2021]	R(2+1)D	112 × 112	K-400	200	**81.1**	44.6
14	CEP [45] [BMVC, 2021]	R(2+1)D	224 × 224	K-400	50	76.7	37.6
15	Ours	R(2+1)D	112 × 112	K-400	20	77.6	40.4
16	VCOP [43] [CVPR, 2019]	R(2+1)D	112 × 112	K-100	200	71.4	32.1
17	VCOP [43] + auxSKD	R(2+1)D	112 × 112	K-100	200	72.6	32.5
18	VideoPace [42] [ECCV, 2020]	R(2+1)D	112 × 112	K-100	18	73.8	36.2
19	VideoPace [42] + auxSKD	R(2+1)D	112 × 112	K-100	18	76.1	38.6
20	RSPNet [5] [AAAI, 2021]	R(2+1)D	112 × 112	K-100	200	74.7	37.4
21	RSPNet [5] + auxSKD	R(2+1)D	112 × 112	K-100	200	75.5	39.0
22	Ours	R(2+1)D	112 × 112	K-100	20	**76.3**	**39.6**
23	VCOP [43] [CVPR, 2019]	R3D-18	112 × 112	K-100	200	58.2	25.2
24	VCOP [43] + auxSKD	R3D-18	112 × 112	K-100	200	60.7	28.4
25	VideoPace [42] [ECCV, 2020]	R3D-18	112 × 112	K-100	18	57.5	23.5
26	VideoPace [42] + auxSKD	R3D-18	112 × 112	K-100	18	60.9	27.1
27	RSPNet [5] [AAAI, 2021]	R3D-18	112 × 112	K-100	200	60.2	32.6
28	RSPNet [5] + auxSKD	R3D-18	112 × 112	K-100	200	61.9	**33.4**
29	Ours	R3D-18	112 × 112	K-100	20	**62.9**	33.0

Table 1. Comparative performance results on UCF101 and HMDB51 when pretraining on K400, and most importantly, on the reduced-size dataset K-100 (shaded region) to emphasise the power of our proposed approach. Note auxSKD refers to our proposed auxiliary pretraining stage using similarity-based knowledge distillation.

4.4. Ablation Studies

We perform ablations to establish the effectiveness of our auxiliary pretraining process and our VSPP pretext task.

Effectiveness of auxSKD – We verify the impact of our auxiliary pretraining stage by showing its gains in performance. In Table 2, we present the results of our proposed method on both R(2+1)D and R3D-18 backbones, with and without auxiliary pretraining for UCF101, using K-100 and K-400 for pretraining. It is clear that in each and every case auxSKD causes an increase in performance.

Temperature parameters – We studied the effect of changing temperatures of auxSKD for both teacher and student models and report the results in Table 3. Here we use VSPP as the pretext task for the primary stage.

Ablation on VSPP – Our VSPP pretext task determines both the segment within a clip where there is a speed alteration compared to the natural speed of the rest of the clip and what the speed rate is, effectively parameters λ and ζ. Based on ablation studies in [42], for all the experiments here we consider 4 different speed rates i.e. $Q = 4$, hence $\lambda = \{1, 2, 3, 4\}$.

Table 4 outlines the effect of each sub-task in VSPP when our model pretrains on them separately and jointly

Method	Pretrain Dataset	UCF101 Top-1	HMDB51 Top-1
Backbone: R(2+1)D			
Ours - auxSKD	UCF101	76.0	37.4
Ours	UCF101	**77.3**	**38.6**
Ours - auxSKD	K-100	74.0	37.3
Ours	K-100	**76.3**	**39.6**
Backbone: R3D-18			
Ours - auxSKD	K-400	65.8	28.8
Ours	K-400	**67.9**	**32.6**
Ours - auxSKD	K-100	60.8	26.3
Ours	K-100	**62.9**	**33.0**

Table 2. Ablation of the auxiliary pretraining stage auxSKD with our proposed approach.

$\gamma^{\mathcal{T}}$	0.01	0.02	0.05	0.07	0.1	0.01	0.02
$\gamma^{\mathcal{S}}$	0.01	0.02	0.05	0.07	0.1	0.1	0.1
UCF101	75.0	**76.3**	75.3	74.9	74.9	75.5	75.0

Table 3. Effect of changing the temperatures for our method for UCF101 with R(2+1)D backbone. $\gamma^{\mathcal{T}}$ and $\gamma^{\mathcal{S}}$ indicate teacher and student temperatures respectively.

(on K-100). Our auxSKD pretraining is not engaged for this ablation. The best result is obtained at 60.8% on the

Speed Prediction	Segment Prediction	#Classes #speed, #segment	UCF101
✓	-	$[Q = 4, Z = 1]$	57.5
✓	✓	$[Q = 4, Z = 2]$	57.5
✓	✓	$[Q = 4, Z = 3]$	59.5
✓	✓	$[Q = 4, Z = 4]$	**60.8**
✓	✗	$[Q = 4, Z = 4]$	58.3
✗	✓	$[Q = 4, Z = 4]$	59.9

Table 4. Ablation of our VSPP pretext task pretrained on K-100 with R3D-18 (no auxSKD stage). We examine the importance of each subtask within VSPP while the number of segments within the clip changes.

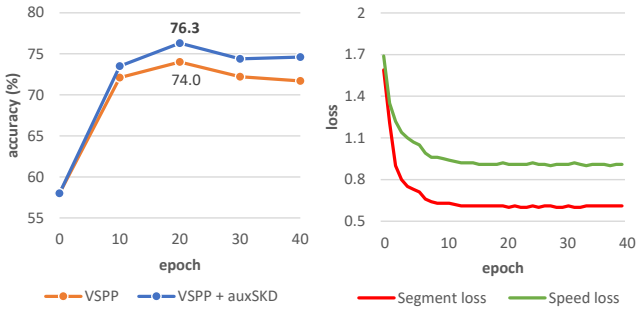

Figure 4. (Left) VSPP pretext task performance with auxSKD (VSPP+auxSKD) and without (VSPP) based on the number of epochs. We pretrained the R(2+1)D model on K-100 for 40 epochs and report the results every 10 epochs on UCF101. (Right) Pretraining losses of our VSSP subtasks on K-100, i.e. speed prediction and segment prediction losses, further illustrate that our model actually converges after around 20 epochs.

UCF101 dataset when pretraining jointly on both tasks and having the maximum number of segments $Z = 4$.

When the number of segments Z during sampling is fewer (i.e. as ζ ranges from 1 to Z) or a subtask is missed out, the performance drops. Note, the first line of the table when there is only one segment, i.e. $Z = 1$ is the equivalent to VideoPace. We believe that increasing the number of segments in the clip pushes the model to temporally explore the video more to find that specific segment with different speed, resulting in better temporal representation.

Pretraining epochs – In Figure 4 (left), we evaluate the performance of our method on UCF101 when pretrained on K-100, with and without auxSKD, using different checkpoints. It can be seen that when auxiliary learning is switched on, the performance of our VSPP pretext task is increased at all checkpoints. We also notice that the performance starts to saturate after 20 epochs for both VSPP and VSPP+auxSKD. In Figure 4 (right), we can see that after around 20 epochs, the changes in the VSPP losses are not significant. This demonstrates that our model converges quickly and we can ensure its convergence during pretraining with only 20 epochs.

5. Limitations

We identify three limitations of our work:

(i) a fundamental aspect of our VSPP pretext task is that it thrives on the altered natural pace of motion in a segment of a video while the rest of the clip retains its natural motion. However, any sudden and very fast motion in a clip may violate this assumption as the fast motion within the selected segment of a clip may be missed when it's sampled. This is a similar limitation for other current speed based pretext tasks such as VideoPace and RSPNet. (ii) we aim to test our approach on at least one other reduced-size version of an existing, large dataset, such as FineAction [23] to further validate our auxiliary pretraining stage as an approach that reduces the dependence of self-supervised learning approaches on large pretraining datasets (iii) other speed-related pretext tasks, such as ASCNet, RSPNet, and VideoPace include an appearance stream in their methodology, however in this work, while the absence of an appearance stream may seem to be a limitation, it was avoided to focus on the power of VSPP as an independent pretext task and promote the auxiliary pretraining stage as two contributions that may be used in a modular fashion by the community. We expect that adding an appearance stream to our model may improve our results.

6. Conclusions

In this paper, we introduced an auxiliary-learning phase for self-supervised video representation learning that allows a significant reduction in the amount of unlabelled data required for the pretraining task. The approach exploits similarity-based knowledge distillation to better prepare a (student) network to perform its primary pretraining task. Our experiments show that this new auxiliary phase auxSKD improves the performance of other existing SSL approaches, such as VCOP [43], VideoPace [42], and RSPNet [5]. We also introduced a new video speed analysis task, VSPP, that predicts the index and altered speed of a segment within a clip which is sampled at a different frame rate to the rest of the clip. Solving this task can strength the network's awareness of the video's natural speed rate and alleviate the imprecise video speed labeling problem [5]. Our experiments illustrate that the features learnt achieve competitive or superior results compared to the state of the art, while training on a much smaller dataset, e.g. K-100 rather than K-400, and at a lower computational cost.

Acknowledgements

The authors are sincerely grateful for the kind donations to the Southmead Hospital Charity and from Caroline Belcher. Their generosity has made this research possible.

References

[1] Soroush Abbasi Koohpayegani, Ajinkya Tejankar, and Hamed Pirsiavash. Compress: Self-supervised learning by compressing representations. *Advances in Neural Information Processing Systems*, 33:12980–12992, 2020. 2, 3, 4

[2] Humam Alwassel, Dhruv Mahajan, Bruno Korbar, Lorenzo Torresani, Bernard Ghanem, and Du Tran. Self-supervised learning by cross-modal audio-video clustering. *Advances in Neural Information Processing Systems*, 33, 2020. 6

[3] Sagie Benaim, Ariel Ephrat, Oran Lang, Inbar Mosseri, William T Freeman, Michael Rubinstein, Michal Irani, and Tali Dekel. Speednet: Learning the speediness in videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9922–9931, 2020. 2, 3, 4, 7

[4] Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541, 2006. 2

[5] Peihao Chen, Deng Huang, Dongliang He, Xiang Long, Runhao Zeng, Shilei Wen, Mingkui Tan, and Chuang Gan. Rspnet: Relative speed perception for unsupervised video representation learning. In *AAAI Conference on Artificial Intelligence*, volume 1, 2021. 1, 2, 3, 4, 5, 6, 7, 8

[6] Dave Epstein, Boyuan Chen, and Carl Vondrick. Oops! predicting unintentional action in video. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 919–929, 2020. 3

[7] Zhiyuan Fang, Jianfeng Wang, Lijuan Wang, Lei Zhang, Yezhou Yang, and Zicheng Liu. Seed: Self-supervised distillation for visual representation. In *International Conference on Learning Representations*, 2021. 2, 3, 4

[8] Basura Fernando, Hakan Bilen, Efstratios Gavves, and Stephen Gould. Self-supervised video representation learning with odd-one-out networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3636–3645, 2017. 1, 2

[9] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Pires, Zhaohan Guo, Mohammad Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. In *Neural Information Processing Systems*, 2020. 1, 4

[10] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 6546–6555, 2018. 6

[11] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020. 1, 4

[12] Olivier J Hénaff, Skanda Koppula, Jean-Baptiste Alayrac, Aaron van den Oord, Oriol Vinyals, and João Carreira. Efficient visual pretraining with contrastive detection. *arXiv preprint arXiv:2103.10957*, 2021. 2

[13] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 2, 3

[14] Deng Huang, Wenhao Wu, Weiwen Hu, Xu Liu, Dongliang He, Zhihua Wu, Xiangmiao Wu, Mingkui Tan, and Errui Ding. Ascnet: Self-supervised video representation learning with appearance-speed consistency. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8096–8105, October 2021. 3, 6, 7

[15] Simon Jenni, Givi Meishvili, and Paolo Favaro. Video representation learning by recognizing temporal transformations. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVIII 16*, pages 425–442. Springer, 2020. 2, 3

[16] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017. 1, 5

[17] Nikos Komodakis and Spyros Gidaris. Unsupervised representation learning by predicting image rotations. In *International Conference on Learning Representations (ICLR)*, 2018. 1

[18] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. Hmdb: a large video database for human motion recognition. In *2011 International conference on computer vision*, pages 2556–2563. IEEE, 2011. 5

[19] Hsin-Ying Lee, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. Unsupervised representation learning by sorting sequences. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 667–676, 2017. 7

[20] Xingyu Lin, Harjatin Singh Baweja, George Kantor, and David Held. Adaptive auxiliary task weighting for reinforcement learning. *Advances in neural information processing systems*, 32, 2019. 2

[21] Yuanze Lin, Xun Guo, and Yan Lu. Self-supervised video representation learning with meta-contrastive network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8239–8249, 2021. 2

[22] Shikun Liu, Andrew J Davison, and Edward Johns. Self-supervised generalisation with meta auxiliary learning. *arXiv preprint arXiv:1901.08933*, 2019. 2

[23] Yi Liu, Limin Wang, Xiao Ma, Yali Wang, and Yu Qiao. Fineaction: A fined video dataset for temporal action localization. *arXiv preprint arXiv: 2105.11107*, 2021. 8

[24] Dezhao Luo, Chang Liu, Yu Zhou, Dongbao Yang, Can Ma, Qixiang Ye, and Weiping Wang. Video cloze procedure for self-supervised spatio-temporal learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11701–11708, 2020. 2

[25] Ishan Misra, C Lawrence Zitnick, and Martial Hebert. Shuffle and learn: unsupervised learning using temporal order verification. In *European Conference on Computer Vision*, pages 527–544. Springer, 2016. 7

[26] Aviv Navon, Idan Achituve, Haggai Maron, Gal Chechik, and Ethan Fetaya. Auxiliary learning by implicit differentiation. *arXiv preprint arXiv:2007.02693*, 2020. 2

[27] Mehdi Noroozi, Ananth Vinjimoor, Paolo Favaro, and Hamed Pirsiavash. Boosting self-supervised learning via knowledge transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9359–9367, 2018. 3

[28] Tian Pan, Yibing Song, Tianyu Yang, Wenhao Jiang, and Wei Liu. Videomoco: Contrastive video representation learning with temporally adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11205–11214, 2021. 1, 6, 7

[29] Wonpyo Park, Dongju Kim, Yan Lu, and Minsu Cho. Relational knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3967–3976, 2019. 3

[30] Baoyun Peng, Xiao Jin, Jiaheng Liu, Dongsheng Li, Yichao Wu, Yu Liu, Shunfeng Zhou, and Zhaoning Zhang. Correlation congruence for knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5007–5016, 2019. 3

[31] Colorado J Reed, Xiangyu Yue, Ani Nrusimha, Sayna Ebrahimi, Vivek Vijaykumar, Richard Mao, Bo Li, Shanghang Zhang, Devin Guillory, Sean Metzger, et al. Self-supervised pretraining improves self-supervised pretraining. *arXiv preprint arXiv:2103.12718*, 2021. 2

[32] Baifeng Shi, Judy Hoffman, Kate Saenko, Trevor Darrell, and Huijuan Xu. Auxiliary task reweighting for minimum-data learning. In *NeurIPS*, 2020. 2

[33] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012. 5

[34] Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for modern deep learning research. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 13693–13696, 2020. 2

[35] Ajinkya Tejankar, Soroush Abbasi Koohpayegani, Vipin Pillai, Paolo Favaro, and Hamed Pirsiavash. Isd: Self-supervised learning by iterative similarity distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9609–9618, 2021. 2, 3, 4

[36] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *International Conference on Learning Representations*, 2019. 2, 3, 4

[37] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pages 776–794. Springer, 2020. 2

[38] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 6450–6459, 2018. 6

[39] Frederick Tung and Greg Mori. Similarity-preserving knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1365–1374, 2019. 2, 3

[40] Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. Simultaneous deep transfer across domains and tasks. In *Proceedings of the IEEE international conference on computer vision*, pages 4068–4076, 2015. 2

[41] Guangting Wang, Yizhou Zhou, Chong Luo, Wenxuan Xie, Wenjun Zeng, and Zhiwei Xiong. Unsupervised visual representation learning by tracking patches in video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2563–2572, 2021. 2

[42] Jiangliu Wang, Jianbo Jiao, and Yun-Hui Liu. Self-supervised video representation learning by pace prediction. In *European Conference on Computer Vision*, pages 504–521. Springer, 2020. 1, 2, 3, 4, 5, 6, 7, 8

[43] Dejing Xu, Jun Xiao, Zhou Zhao, Jian Shao, Di Xie, and Yueting Zhuang. Self-supervised spatiotemporal learning via video clip order prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10334–10343, 2019. 2, 6, 7, 8

[44] Guodong Xu, Ziwei Liu, Xiaoxiao Li, and Chen Change Loy. Knowledge distillation meets self-supervision. In *European Conference on Computer Vision*, pages 588–604. Springer, 2020. 2

[45] Xinyu Yang, Majid Mirmehdi, and Tilo Burghardt. Back to the future: Cycle encoding prediction for self-supervised contrastive video representation learning. *arXiv preprint arXiv:2010.07217*, 2020. 7

[46] Yuan Yao, Chang Liu, Dezhao Luo, Yu Zhou, and Qixiang 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*, pages 6548–6557, 2020. 3

[47] Junho Yim, Donggyu Joo, Jihoon Bae, and Junmo Kim. A gift from knowledge distillation: Fast optimization, network minimization and transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4133–4141, 2017. 3

[48] Sergey Zagoruyko and Nikos Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. *arXiv preprint arXiv:1612.03928*, 2016. 3

[49] Xiaohua Zhai, Avital Oliver, Alexander Kolesnikov, and Lucas Beyer. S4l: Self-supervised semi-supervised learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1476–1485, 2019. 2