Self-supervised Video Representation Learning with Cascade Positive Retrieval

Cheng-En Wu¹*, Farley Lai², Yu Hen Hu¹, Asim Kadav²
¹ Department of Electrical and Computer Engineering, University of Wisconsin-Madison, WI, USA
² NEC Laboratories America, Inc., San Jose, CA, USA
{cwu356, yhhu}@wisc.edu
farleylai@icloud.com asimkadav@gmail.com

Abstract

Self-supervised video representation learning has been shown to effectively improve downstream tasks such as video retrieval and action recognition. In this paper, we present the Cascade Positive Retrieval (CPR) that successively mines positive examples w.r.t. the query for contrastive learning in a cascade of stages. Specifically, CPR exploits multiple views of a query example in different modalities, where an alternative view may help find another positive example dissimilar in the query view. We explore the effects of possible CPR configurations in ablations including the number of mining stages, the top similar example selection ratio in each stage, and progressive training with an incremental number of the final Top-k selection. The overall mining quality is measured to reflect the recall across training set classes. CPR reaches a median class mining recall of 83.3%, outperforming previous work by 5.5%. Implementation-wise, CPR is complementary to pretext tasks and can be easily applied to previous work. In the evaluation of pretraining on UCF101, CPR consistently improves existing work and even achieves state-of-the-art R@1 of 56.7% and 24.4% in video retrieval as well as 83.8% and 54.8% in action recognition on UCF101 and HMDB51. The code is available at https://github.com/necla-ml/CPR.

1. Introduction

Recently, large-scale self-supervised pretraining such as BERT [7] and DINO [3] has been shown to improve the representations and potentially outperform its supervised counterpart. Most approaches revolve around proposing pretext tasks [1, 10, 16, 19, 23, 38, 40–42] based on instance discrimination to learn representations by matching or classifying specific relationships between the query example and its augmented variants with the objective to minimize the contrastive loss [33] and other predictive losses. However, few address the lack of true positives (TP) other than the query example variants and likely harmful false negatives uniformly sampled from the entire dataset [6]. Previous work CoCLR [12] demonstrates the significant performance gap with the upper bound achieved in a supervised contrastive setting using the labels for TP as in [18].

We are inspired by related work [11, 14, 27, 32, 35, 37] that exploits multi-views of video to learn the representations through the correspondences between different modalities. Previous work [12, 17, 30] incorporating hard example mining in metric learning, object detection and action recognition further motivates the necessity of positive example mining in self-supervised representation learning. As for video representation learning, hard positive examples in the RGB view may be mined from the motion view despite seemingly different background appearances. On the other hand, hard positive examples in the motion view may be mined from the RGB view as the motions can differ significantly from various camera angles while the background remains similar in the RGB view for actions in the same class. CoCLR [12] shows mining in the alternative view during training improves the representations and downstream task performance. Nonetheless, it is not necessarily sufficient for mining only once in a single view to prevent sampling false positives (FP).

To address this issue, we propose the Cascade Positive Retrieval (CPR) and systematically explore the design space of positive example mining. The idea is to refine the mining successively in a cascade of stages across different views as search with filters to be applied progressively. For instance, given a query example, one may first select those with similar background in the RGB view, then further filter out those dissimilar in the motion view and so on. Apparently, the number of mining stages and the selection ratio in each stage matter. The goal is to conclude the strategy for effective positive example mining and make it applicable to existing work. Moreover, it remains unclear of the overall mining quality in terms of the recall across train-
ing set classes despite the R@1 mining retrieval recall by CoCLR [12]. We measure and compare the mining quality that suggests correlation with the resulting performance in ablations.

In short, we make the following contributions:

1.) We propose the Cascade Positive Retrieval for self-supervised learning (SSL) of video representations that complements pretext tasks and can be applied to existing work easily regardless of the SSL framework used.

2.) We apply CPR to previous work and observed consistent improvement in downstream video retrieval and action recognition. We then extensively explore the design space of mining configurations in ablations w.r.t. the number of stages in the cascade, the top similar example selection ratio in each stage and the progressive training regime.

3.) We measure the mining quality of CPR in terms of the positive mining recall denoting each time the fraction of TPs in the final stage Top-k selected as the positive set, and the class mining recall representing the fraction of distinct TPs selected from a class in one training epoch.

4.) We evaluate the transfer performance in video retrieval and action recognition on UCF101 and HMDB51 from pretraining on UCF101 with CPR applied to an existing work, achieving state-of-the-art (SOTA) results.

2. Related Work

Self-supervised Learning. Large-scale representation learning through self-supervision has achieved great success in multiple fields including natural language processing (NLP) and computer vision (CV). In NLP, the general idea is to build a language model that learns to predict masked out words as in BERT [7]. In CV, the feature extraction backbone is trained to learn representations based on instance discrimination that works on both images and videos. The instance discrimination views an example and its augmented variants as positive while the other examples are treated as negative. A typical objective is to minimize the contrastive loss that encourages positive examples to be similar in representations while pushing away negative examples. Many SSL frameworks were proposed in recent years such as SimCLR [4], BYOL [9], MoCo [5, 13] and SwAV [2] to facilitate systematic composition of numerous pretext tasks that augment the input examples and formulate the contrastive loss, delivering competitive performance in comparison with supervised counterparts. In this paper, we focus on improving self-supervised video representation learning from the perspective of hard positive example mining and show our method can be easily applied to existing work regardless of a particular SSL framework used or not.

Video Representation Learning. In contrast with SSL of images, videos enables rich spatiotemporal augmentation to generate diverse positive and negative example clips from sampled frames. Common pretext tasks include future prediction [11] and speed prediction [1, 16, 38, 42] to infer the relationship between clips and the pace a clip is sampled. Other tasks may require to sort out the ordering of frames or clips [41], solve jigsaw puzzles [19], match features in different modalities [14, 27, 35, 37] or group visual entities based on co-occurrences in space and time [15]. We target the video domain as videos in multiple views potentially provide opportunities to mine hard positive examples in the query class. Nonetheless, the proposed method is not limited to video tasks or specific pretext tasks. Instead, we aim to complement existing approaches with hard positive example mining.

Hard Example Mining. Hard example mining in supervised learning is well studied in metric learning and other CV tasks. In metric learning, the goal is the push away those hard negative examples but the challenge is the intractable computational overhead over large datasets as the embedding is updated constantly. One possible solution is to efficiently sample negative instances in nearest classes as in deep metric learning [30]. Regarding positive example mining, InP [34] selects positive examples that preserve high semantic consistency through a recursive k-nearest neighbors graph. In addition, CMA [25] introduces the cross-modal agreement that discovers positive examples highly similar in both audio and visual feature space through multi-view learning [32].

In video object detection, [17] leverages the temporal consistency to identify hard negative and positive examples from detection misses and isolated detection in consecutive video frames.

In the case of SSL, it is challenging for no labels and the representation learning is limited to the augmentations of the query example with instance discrimination for the lack of hard positive examples in the query class. Worse, the negative examples are uniformly sampled and potentially include false negatives (FN). This is called the sampling bias in [6] and a possible solution is to reweight the positive and negative terms in the contrastive loss for correction given the estimated class priors [6, 28].

On the other hand, as with the video object detection, self-supervised video representation learning may exploit multi-views of video clips to mine hard positive examples. CoCLR [12] mines positive examples from action recognition datasets given a query example in the RGB view with its corresponding motion or flow view. Intuitively, this may help find positive examples with similar motions despite dissimilar background and vice versa. Our work further explores the possibilities to mine diverse positive examples in the query example class as CoCLR only mines
positive examples in one view at a time. Chances are out of those with similar motions, top instances similar in the RGB view could be more likely the true positives. Therefore, we reshape the positive example mining as a cascade refining process between different video views. While CoCLR measures R@1 for mining retrieval recall, we further evaluate the mining quality in terms of the overall mining recall across the classes throughout training in reflection of the coverage of distinct class instances. The metric is expected to correlate with the resulting performance w.r.t. the upper bound in the supervised contrastive setting where the mining recall is essentially perfect for all the class instances being selected during training.

3. Proposed Method

Algorithm 1 CPR: Cascade Positive Retrieval

| Variables: MB, C, S, B, V, r, v, q, q+, q− |

1. C ⊆ Z+ \(\triangleright\) range of training cycles
2. S ⊆ Z+ \(\triangleright\) epochs given a training cycle
3. K(c, s) \(\in\) Z+ \(\triangleright\) Top-k to select at stage s in cycle c
4. v \(\in\) B \(\triangleright\) selection ratio before the last stage
5. select \((f_c, \text{candidates}_{s, r})\), topk \((f_c, \text{candidates}_{s, r})\) \(\in\) Z+ \(\triangleright\) select top similar instances by ratio
6. SV(s) \(\in\) Z+ \(\triangleright\) given a view at stage s
7. V \(\in\) \{v1, v2, \ldots\} \(\triangleright\) a set of views

8. for \(v \in\) V do
9.   for \(q \in\) V do
10.  if \(v == v\) then
11.    \(f\) \(\leftarrow\) encoder\(_v\)(q\(_v\))
12.    \(f\) \(\leftarrow\) encoder\(_v\)(q\(_-\))
13.  else
14.    \(f\) \(\leftarrow\) encoder\(_v\)(\(\bar{f}\))
15.  end if
16. end for
17. end for
18. end for
19. end for
20. end for
21. end for

In this section, we revisit the concept of contrastive learning with different discrimination learning objectives. Next, we present CPR in Algorithm 1, detailing the cascade positive retrieval for mining examples in general.

3.1. Instance Discrimination

Self-supervised video representation learning based instance discrimination where each instance serves as its own class has been shown effective with the contrastive loss of InfoNCE [33]. Specifically, given a set of videos \(V\), a video clip \(v_i\) is a number of frames sampled from a video in \(V\) and its positive variant \(v_i^+\) that can be an augmentation or another clip sampled from the same video, forming a positive pair \((v_i, v_i^+)\). On the other hand, a set of negative examples \(N^-\) consists of those clips \(v_j^-\), \(j \neq i\). These clips are fed into a query encoder and a key encoder to obtain the visual representations. The output features of the query, its positive augmentation and negative keys are denoted by \(q_i\), \(q_i^+\), and \(k_j^-\) respectively. The InfoNCE loss is defined as follows:

\[
\mathcal{L}_N = -\log \frac{\exp(q_i \cdot q_i^+ / \tau)}{\sum_{p \in P} \exp(q_i \cdot q_i^+ / \tau) + \sum_{j=1}^{N^-} \exp(q_i \cdot k_j^- / \tau)}
\]

where the similarity is measured by dot product with a temperature hyperparameter \(\tau\) to adjust its scale. Intuitively, InfoNCE encourages to pull positive pairs closer while pushing away negative pairs.

3.2. Multi-instance Discrimination

In the case of multiple positive pairs, Multi-Instance InfoNCE or MIL-NCE proposed in [24] is defined as follows:

\[
\mathcal{L}_M = -\log \frac{\sum_{p \in P} \exp(q_i \cdot q_i^+ / \tau)}{\sum_{p \in P} \exp(q_i \cdot q_i^+ / \tau) + \sum_{j=1}^{N^-} \exp(q_i \cdot k_j^- / \tau)}
\]

where \(P\) is a positive set containing positive augmentation of the query and other keys with the same label as the query. For example, in an action video dataset, a fencing positive set includes the augmentation of the query video and other videos with the fencing label.

3.3. Cascade Positive Retrieval

In view of issues with instance discrimination including the lack of other non-augmented positives and potential false negatives, previous work CoCLR [12] has proposed to mine positive examples in an alternative view other than the query view. However, there is a possibility that CoCLR suffers from FPs with similar motion patterns from the flow view because the mining in the alternative view is only done once such that some actions with very similar motion patterns such as Showout and ThrowDiscus may be wrongly selected and confuse the model as shown in Figure 2. Unlike CoCLR mining heavily dependent on a single view, our CPR fully exploits the advantage of multi-views to improve the mining quality. Figure 1 illustrates that in one cascade of positive retrieval, CPR alternates between the RGB and flow views to mine a top number of positive examples with most similar appearances and motions as the query clip. When applying CPR to existing work, there are many possible configurations and hyperparameters to consider as described in Algorithm 1 that assumes a memory bank \(MB\) storing encoded instance features in different views, a progressive training schedule in cycles, the number of epochs.
in one cycle, the number of mining stages in one cascade, the selection ratio of top similar examples at each stage and etc. Specifically, the algorithm iterates through each cycle \( c \) and epoch \( e \) to train with query examples in batches \( B \). Each batch consists of query examples and their positive variants from augmentation or sampling as \( q \) and \( q^+ \). In the beginning of the batch processing, the representation encoder to train in the query view, \( \text{encoder}_{v_q} \), encodes the query examples \( q_{v_q} \) and produces the features \( f_{q_{v_q}} \). Those \( q^+ \) may be encoded in the query view with the momentum encoder \( \text{encoder}_{v_q}^{ema}(q_{v_q}^+) \) and in the other views with frozen encoder \( \text{encoder}_{v_q}^{fixed}(q_{v_q}^+) \). Next, CPR retrieves the most similar examples in successive stages \( S \) given a selection ratio \( r \) used at stages before the last one and a Top-\( k \) for the final stage selection determined by the current cycle \( c \) and stage \( s \). Note that the mining always uses the features of positive query variants to measure the similarities with those stored in \( MB \) by view. Eventually, a set of Top-\( k \) most likely positives are selected at the last stage as \( \text{pos} \) and combined with \( q^+ \). The other instances in \( MB \) are viewed as negatives \( \text{neg} \). Then the MIL-NCE loss is computed given the query examples, mined positives and negatives to optimize the encoder in the query view. Afterwards, the memory bank \( MB \) is updated with the newly encoded query example variants for the next batch training iteration. In the next section, we will evaluate the effects of changing CPR hyperparameters in ablations as well as compare the performance with SOTAs.

4. Experiments

4.1. Setup

Dataset. In this section, we conduct ablation studies and evaluate CPR on two action video datasets:

**UCF101** [29] contains 13K videos in 101 human action classes at more diverse camera angles than HMDB51. Out of the three splits of the dataset, the first one is used for our ablations, pretraining, and downstream task evaluations.

**HMDB51** [21] consists of 7K videos in 51 human action categories. The dataset is divided into three splits. We use the first split to conduct two downstream tasks in video retrieval and action recognition.

Implementations. We apply CPR to previous work IIC [31] and CoCLR [12]. While the latter uses MoCo [13], CPR is not dependent on specific SSL frameworks. For fair comparison, we use exactly the same hyperparameters as previous work and only plug in CPR to construct the positive and negative sets for computing the MIL-NCE loss. If necessary, we even retrain previous work for the same number of epochs to compare with the reproduced results. More details can be found in the supplemental materials.

Data Preprocessing: The data preparation follows previous work respectively. As for CoCLR [12], a clip in both RGB and flow views is randomly sampled from 32 consecutive frames in the video. Each frame is randomly cropped and resized to 128×128 pixels. We apply the same data augmentations including horizontal flips, color jittering and Gaussian blur to the clips. Note that Gaussian blur is not used for downstream tasks. To generate optical flow maps from

Figure 1. Overview of CPR in mining alternately from both RGB and flow views for one cascade retrieval. \( K \) denotes the size of the memory queues storing instance features in both views. Given a query example in the RGB view, the mining starts with selecting top \( k_1 \) most similar instances from the flow memory queue at stage 1, top \( k_2 \) most similar instances from the RGB memory queue at stage 2, and so on up to top \( k_{n-1} \) from the RGB memory queue at stage \( n-1 \), where \( k_1, k_2, \ldots, k_{n-1} \) are values derived from the number of instances selected from the previous stage multiplied by a fixed selection ratio (e.g. 0.5) at each stage. Unlike previous stages, the Top-\( k_n \) most similar instances at the final stage are selected to form the positive set.
the video, we use TV-L1 [43] to extract the flow view with a third channel filled with zeros. The features are clipped in the range of 20 pixels and rescaled from $[-20, 20]$ to $[0, 255]$. In contrast, the motion view for IIC [31] is based on frame difference residuals.

**Self-supervised Pretraining on UCF101.** For IIC [31], we train from scratch with CPR under NPID [39]. For CoCLR [12], we begin with the released RGB and flow models pretrained with InfoNCE as there is no positive mining in the initialization. Next at the co-training stage, the RGB and flow models are alternately trained for 400 epochs on 2 GPUs, each with a batch size of 16. That is the same number of epochs as two cycles in CoCLR.

**Video Retrieval.** We evaluate video retrieval as a downstream task on both UCF101 and HMDB51 based on extracted features from the pretrained model without finetuning. Following the test protocol in [23, 41], we take a video in the test set as a query and use it to retrieve $k$-nearest neighbors in its corresponding training set. The recall at $k$ ($R@k$) serves as the evaluation metric, which means if one of the retrieved top $k$ nearest neighbors is from the same class as the query, it is counted as a correct retrieval result.

**Action Recognition.** In addition to video retrieval, we also evaluate the action recognition performance of the pretrained models on UCF101 and HMDB51. The pretrained models are transferred as the feature extraction backbone for downstream tasks. Two scenarios including linear probing and finetuning are considered respectively. For linear probing, we freeze the backbone while training the linear classifier only. For finetuning, we train the entire network including the backbone and the linear classifier. The training and evaluation protocols essentially follow previous work for fair comparison even with test time augmentation used.

### 4.2. Ablation Study

In this section, we explore CPR in numerous configurations. All experiments are conducted on UCF101 following the setup mentioned in Section 4.1 except for the number of training epochs fixed at 100 for pretraining and finetuning respectively. Unless said otherwise, the ablations are based

<table>
<thead>
<tr>
<th>Stages $(s)$</th>
<th>$R@1$</th>
<th>$R@5$</th>
<th>$R@10$</th>
<th>Probe</th>
<th>Finetune</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s = 1$</td>
<td>45.1</td>
<td>64.0</td>
<td>71.9</td>
<td>60.0</td>
<td>69.5</td>
</tr>
<tr>
<td>$s = 3$</td>
<td>46.5</td>
<td>64.5</td>
<td>72.0</td>
<td>60.0</td>
<td>69.6</td>
</tr>
<tr>
<td>$s = 5$</td>
<td>47.5</td>
<td>65.1</td>
<td>73.3</td>
<td>60.2</td>
<td>70.6</td>
</tr>
<tr>
<td>$s = 7$</td>
<td>47.8</td>
<td>66.2</td>
<td>74.6</td>
<td>60.4</td>
<td>71.3</td>
</tr>
</tbody>
</table>

Table 1. Ablations with CPR applied to CoCLR w.r.t. the number of stages. CoCLR is a special case with CPR in only one stage as $s = 1$ where only the Top-5 positive candidates are selected.

<table>
<thead>
<tr>
<th>SR $(r)$</th>
<th>$R@1$</th>
<th>$R@5$</th>
<th>$R@10$</th>
<th>Probe</th>
<th>Finetune</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r = 0.8_{(s=3)}$</td>
<td>46.1</td>
<td>63.9</td>
<td>72.4</td>
<td>60.0</td>
<td>69.8</td>
</tr>
<tr>
<td>$r = 0.5_{(s=3)}$</td>
<td>46.5</td>
<td>64.5</td>
<td>72.0</td>
<td>60.0</td>
<td>69.6</td>
</tr>
<tr>
<td>$r = 0.8_{(s=7)}$</td>
<td>46.2</td>
<td>63.6</td>
<td>71.7</td>
<td>59.6</td>
<td>69.9</td>
</tr>
<tr>
<td>$r = 0.5_{(s=7)}$</td>
<td>47.8</td>
<td>66.2</td>
<td>74.6</td>
<td>60.4</td>
<td>71.3</td>
</tr>
</tbody>
</table>

Table 2. Results for CPR applied to CoCLR with varied selection ratios but a fixed number of stages $s$.

<table>
<thead>
<tr>
<th>IIC(+CPR)</th>
<th>$R@1$</th>
<th>$R@5$</th>
<th>$R@10$</th>
<th>R@20</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline-Top-5</td>
<td>36.2</td>
<td>53.5</td>
<td>63.6</td>
<td>72.7</td>
</tr>
<tr>
<td>Prog-Top-1</td>
<td>39.4</td>
<td>57.5</td>
<td>67.6</td>
<td>77.2</td>
</tr>
<tr>
<td>Prog-Top-2</td>
<td>42.5</td>
<td>60.5</td>
<td>69.0</td>
<td>77.5</td>
</tr>
<tr>
<td>Prog-Top-3</td>
<td>44.1</td>
<td>62.3</td>
<td>70.2</td>
<td>77.9</td>
</tr>
<tr>
<td>Prog-Top-4</td>
<td>45.3</td>
<td>63.2</td>
<td>70.4</td>
<td>78.2</td>
</tr>
<tr>
<td>Prog-Top-5</td>
<td>46.2</td>
<td>63.6</td>
<td>71.4</td>
<td>79.2</td>
</tr>
</tbody>
</table>

Table 3. Improvements in video retrieval with progressive training when CPR is applied to IIC [31] in 5 cycles with incremental Top-$k$ selection. The baseline is trained with the same number of total epochs in the 5 cycles with a fixed Top-$k$ selection at the last stage.

<table>
<thead>
<tr>
<th>Settings</th>
<th>PMR</th>
<th>$R@1$</th>
<th>Finetune</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s = 1$ (CoCLR)</td>
<td>35.1</td>
<td>45.1</td>
<td>69.5</td>
</tr>
<tr>
<td>$s = 3$, $r = 0.5$</td>
<td>35.8</td>
<td>46.5</td>
<td>69.6</td>
</tr>
<tr>
<td>$s = 5$, $r = 0.5$</td>
<td>37.4</td>
<td>47.5</td>
<td>70.6</td>
</tr>
<tr>
<td>$s = 7$, $r = 0.5$</td>
<td>38.9</td>
<td>47.8</td>
<td>71.3</td>
</tr>
</tbody>
</table>

Table 4. Mean PMR and $R@1$ measured in the last epoch as well as fine-tuning results w.r.t. different CPR configurations.
Figure 3. Positive Mining Recall (PMR) for 100 training epochs. The results are generated by both models pretrained on UCF101.

Figure 4. TP class instance mining counts in selected classes.

<table>
<thead>
<tr>
<th>Method</th>
<th>R@1</th>
<th>R@5</th>
<th>R@10</th>
<th>Finetune</th>
</tr>
</thead>
<tbody>
<tr>
<td>IIC [31]</td>
<td>34.8</td>
<td>51.6</td>
<td>60.8</td>
<td>71.8</td>
</tr>
<tr>
<td>IIC(+CPR)</td>
<td>46.2</td>
<td>63.6</td>
<td>71.4</td>
<td>73.1</td>
</tr>
<tr>
<td>CoCLR</td>
<td>45.1</td>
<td>64.1</td>
<td>71.9</td>
<td>69.5</td>
</tr>
<tr>
<td>CoCLR(+CPR)</td>
<td>47.8</td>
<td>66.2</td>
<td>74.6</td>
<td>71.3</td>
</tr>
<tr>
<td>UberNCE</td>
<td>70.3</td>
<td>81.7</td>
<td>86.8</td>
<td>80.7</td>
</tr>
</tbody>
</table>

Table 5. Summary of improvements over IIC and reproduced CoCLR [12] with CPR on UCF101. UberNCE is reproduced in the supervised contrastive setting serving as the upper bound.

on application of CPR to CoCLR.

**Number of Stages.** Our CPR mines positive examples in a cascade of multiple stages. It is necessary to demonstrate the influence of this hyperparameter given a fixed selection ratio 0.5 for positive selection across stages before the last one and Top-5 for the last stage. As shown in Table 1, with more mining stages, the model may learn better representations for the downstream task and the best performance is achieved in the configuration with 7 stages. As a result, we use 7 stages in later evaluation with other SOTAs.

**Selection Ratio.** In this ablation, we set the selection ratio (SR) to 0.5 and 0.8 respectively to evaluate the impact of SR before the final stage that uses fixed Top-5. The total number of stages are set to 5 and 7 for comparison. It is observed that no matter in 5 or 7 stages, a smaller SR can get better performance in Table 2. Hence, we choose $SR = 0.5$ when comparing with other SOTAs.

**Progressive Training.** This configuration examines the training regime of the Top-$k$ selection at the last stage. Specifically, is it better to train with a fixed Top-$k$ or the training should be progressive with an incremental number of Top-$k$. The conclusion is likely model architecture dependent as we see the improvement with IIC shown in Table 3 in terms of video retrieval recalls but little with CoCLR. Therefore, progressive training will not be applied to CoCLR in other evaluations.

**CPR Mining Quality.** While CoCLR [12] measures R@1 for the mining retrieval against the ground truth (GT) throughout training, it remains unclear how many TPs are actually mined each time and throughout the training. Therefore, we measure the positive mining recall and class mining recall defined in Eq. 3 and Eq. 4:

$$\text{Positive Mining Recall} = \frac{\# TP}{\text{Size of Positive Set}}$$

$$\text{Class Mining Recall} = \frac{\# \text{Distinct TP Selected}}{\# \text{Total Class Instances}}$$

The positive mining recall (PMR) measures each time the fraction of TPs in the final Top-$k$ selected as the positive
set. The class mining recall (CMR) measures the fraction of distinct TPs selected from a class in one training epoch. Table 4 shows that as PMR and mining R@1 increase with more stages, higher fine-tuning performance on action recognition is expected. However, PMR seems to serve as a better performance indicator for being in proportion to improvement.

Furthermore, we provide a breakdown of full PMR during the entire pretraining process for 100 epochs on UCF101 in Figure 3. The results show CPR gradually increases its PMR from 34.0% to 40.3%. On the other hand, the PMR of baseline CoCLR is sluggish between 35.2% and 37.4%. It can be found that CPR indeed can mine more true positives by leveraging both RGB and flow views while baseline CoCLR suffers from false positives for mining only in a single view. This is the crucial factor to support why CPR has better performance than baseline CoCLR. In summary, PMR seems to serve as a better performance indicator for being in proportion to improvement.

To further quantify the mining quality across classes, we count the number of distinct TPs selected for each action class. Figure 4 illustrates the statistics from 10 randomly chosen classes with 18 instances each in the last training epoch. CPR succeeds in selecting all the distinct TPs from both Surfing and UnevenBars classes while discovering much less from the Hammering class. Through visual inspection, Hammering is difficult with motions at different camera angles in varied scenes. In contrast, Swing and Surfing are easy to mine for having regular motion patterns and consistent background. Furthermore, we list the CMRs of the Top-3 and the bottom 3 classes respectively. The Top-3 classes are Diving(100%), PommelHorse(100%), and UnevenBars(100%). On the other hand, the bottom 3 classes are Lunges(27.8%), Haircut(33.3%) and HandstandWalking(33.3%). We demonstrate video frames from these classes above to visualize their content including human action and background. In Figure 5, sampled frames in the bottom 3 classes cover different camera angles and varied backgrounds, which increases the difficulty of mining full video instances in each class. In contrast, it is simple to discover entire video instances in each Top-3 class because these classes represent a relatively consistent background and standard motion with a fixed pattern. To sum up, out of all the UCF101 classes, CPR scores higher CMR than baseline CoCLR in 48 classes while the baseline mines better only in 20 classes. It is even in the rest classes. Overall, CPR achieves the median CMR of 83.3% across all the classes, which is 5.5% improvement over the baseline CoCLR with the median CMR of 77.8%.

Besides quantitative measurements, we visualize the Top-5 positive examples mined from the baseline CoCLR and CPR for qualitative comparison. In Figure 2, the baseline mining heavily relies on the motions from optical flows and tends to select false positives (FPs) with similar motion patterns to the query. Instead, our CPR alternately mines from both RGB and flow views to discover positive examples with similar appearances and motions to the query. Even the only FP still contains visually similar motions and context as the query. This indicates that CPR is able to effectively filter out potential FPs from a single view. Consequently, CPR facilitates learning representations from more diverse TPs compared with the baseline mining.

Applicability. In addition to CoCLR, CPR is also applied to IIC [31] in the ablation of progressive training, demonstrat-
ing the general applicability. Particularly, IIC uses memory banks instead of momentum encoders to maintain features as well as frame difference residuals as motion views. It focuses on generating hard negatives from the query video by repeating or shuffling the frames but there is no positive example mining. With CPR, IIC gains significant performance improvement in both downstream tasks on UCF101 in Table 5 where CoCLR is also listed to show consistent performance boost. This suggests that CPR is generally applicable whether or not the existing approach has positive example mining in mind.

4.3. Comparison with State-of-the-arts

As CPR aims to benefit existing work in terms of better positive example mining, our focus is to show how much improvement an existing work can be enhanced with CPR to compete with newer SOTAs. CoCLR [12] is chosen as it already has positive example mining in mind.

**Video Retrieval.** To validate the effectiveness of learned representations with CPR, we evaluate the nearest neighbor video retrieval on both UCF101 and HMDB51. Specifically, the top-k video retrieval recalls for $k = 1, 5, 10, 20$ are computed as the performance metrics. As shown in Table 6, CoCLR-Flow(+CPR) outperforms the the other SOTA methods in all recall metrics on both datasets. We achieve the best top-1 recall of 56.7% on UCF101 and 24.4% on HMDB51, outperforming the the latest SOTA MCN [22] by up to 2.9% based on the same backbone. Moreover, CPR also gains much more improvement at higher top-k metrics. Since video retrieval does not require fine-tuning and leaves little room for manipulation, positive example mining from diverse positive examples across distinct videos is likely the key to learning effective representations.

**Action Recognition.** In table 7, we compare our method with SOTAs on video action recognition. All methods are applied in a fully finetuning setting that finetunes all layers on the downstream task. Pretrained on UCF101, CoCLR-Flow(+CPR) outperforms all the previous SOTAs fine-tuned on UCF101 and HMDB51 with accuracies of 83.8% and 54.8% based on the same or comparable backbone and resolutions as illustrated in Table 7.

<table>
<thead>
<tr>
<th>Method</th>
<th>Year</th>
<th>Dataset</th>
<th>Resolution</th>
<th>Architecture</th>
<th>UCF101</th>
<th>HMDB51</th>
</tr>
</thead>
<tbody>
<tr>
<td>VTOP [41]</td>
<td>2019</td>
<td>UCF101</td>
<td>16 × 112²</td>
<td>R(2+1)D-26</td>
<td>72.4</td>
<td>30.9</td>
</tr>
<tr>
<td>VCP [23]</td>
<td>2020</td>
<td>UCF101</td>
<td>16 × 112²</td>
<td>C3D</td>
<td>68.5</td>
<td>32.5</td>
</tr>
<tr>
<td>IIC [31]</td>
<td>2020</td>
<td>UCF101</td>
<td>16 × 112²</td>
<td>R3D-18</td>
<td>74.4</td>
<td>38.3</td>
</tr>
<tr>
<td>PacePred [38]</td>
<td>2020</td>
<td>UCF101</td>
<td>16 × 112²</td>
<td>R(2+1)D</td>
<td>75.9</td>
<td>35.9</td>
</tr>
<tr>
<td>PRP [42]</td>
<td>2020</td>
<td>UCF101</td>
<td>16 × 112²</td>
<td>C3D</td>
<td>69.1</td>
<td>34.5</td>
</tr>
<tr>
<td>TT [16]</td>
<td>2020</td>
<td>UCF101</td>
<td>16 × 112²</td>
<td>R3D-18</td>
<td>77.3</td>
<td>47.5</td>
</tr>
<tr>
<td>CoCLR-RGB</td>
<td>2020</td>
<td>UCF101</td>
<td>32 × 128²</td>
<td>R3D-18</td>
<td>81.4</td>
<td>52.1</td>
</tr>
<tr>
<td>DSM [35]</td>
<td>2021</td>
<td>UCF101</td>
<td>16 × 112²</td>
<td>C3D</td>
<td>70.3</td>
<td>40.5</td>
</tr>
<tr>
<td>STS [36]</td>
<td>2021</td>
<td>UCF101</td>
<td>16 × 112²</td>
<td>R3D-18</td>
<td>77.8</td>
<td>40.7</td>
</tr>
<tr>
<td>CMD [14]</td>
<td>2021</td>
<td>UCF101</td>
<td>16 × 112²</td>
<td>R3D-26</td>
<td>76.6</td>
<td>47.2</td>
</tr>
<tr>
<td>MFO [26]</td>
<td>2021</td>
<td>UCF101</td>
<td>32 × 128²</td>
<td>S3D</td>
<td>74.3</td>
<td>37.2</td>
</tr>
<tr>
<td>ViCLR [8]</td>
<td>2021</td>
<td>UCF101</td>
<td>32 × 128²</td>
<td>S3D</td>
<td>82.8</td>
<td>52.9</td>
</tr>
<tr>
<td>MCN [22]</td>
<td>2021</td>
<td>UCF101</td>
<td>32 × 128²</td>
<td>S3D</td>
<td>82.9</td>
<td>53.8</td>
</tr>
<tr>
<td>CoCLR-Flow(+CPR)</td>
<td>UCF101</td>
<td>32 × 128²</td>
<td>S3D</td>
<td>83.8</td>
<td>54.8</td>
<td></td>
</tr>
</tbody>
</table>

Table 7. Comparison with SOTA action recognition on UCF101 and HMDB51 based on pretraining on UCF101
References


[40] Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In ECCV, 2018. 1

[41] Dejing Xu, Jun Xiao, Zhou Zhao, Jian Shao, Di Xie, and Yueting Zhuang. Self-supervised spatiotemporal learning via video clip order prediction. In CVPR. IEEE, June 2019. 1, 2, 5, 7, 8
