Probabilistic Representations for Video Contrastive Learning -Supplementary Materials-

Jungin Park[1] Jiyoung Lee[2] Ig-Jae Kim[3] Kwanghoon Sohn[1*]
[1]Yonsei University [2]NAVER AI Lab [3]Korea Institute of Science and Technology (KIST)
{newrun, khsohn}@yonsei.ac.kr lee.j@navercorp.com

In this document, we include supplementary materials for ProVico. We first describe methodological details on ProViCo (Sec. A, Sec. B) and provide implementation details (Sec. C) for downstream tasks. The additional experimental results (Sec. D), including ablation studies and qualitative results, are also presented to complement the main paper. Finally, observations and intuitive analyses to leverage the uncertainty for the future work are introduced in Sec. E. For all experiments in this document, we use R(2+1)D [12] as the backbone network.

A. Uncertainty Computation on ProViCo

In ProViCo, the whole video distribution is esimated as a Mixture of Gaussians (MoG) with N clip embeddings. Moreover, the variance predicted for each video represents the inherent uncertainty for the data. To compute the uncertainty of videos, we briefly provide the computations of the mean and variance of the whole video distribution from the clip distribution. Given a set of N clips $\{c_n\}_{n=1}^{N}$ sampled from a video $\mathcal{V}$, let $f_{c_n}(z)$ be a probability density function (PDF) for the n-th clip with a mean vector $\mu_n \in \mathbb{R}^D$ and a diagonal components of covariance matrix $\sigma_n^2 \in \mathbb{R}^D$. The PDF of the MoG is represented by the averaged PDF of N embeddings:

$$f_{\mathcal{V}}(z) = \frac{1}{N}\sum_{n} f_{c_n}(z). \tag{1}$$

Then, the average vector $\mu_{\mathcal{V}}$ for the MoG is computed by the averaged mean vectors from clips:

$$\begin{aligned}\mu_{\mathcal{V}} &= \int z f_{\mathcal{V}}(z)dz \\ &= \frac{1}{N}\sum_{n}\int z f_{c_n}(z)dz \\ &= \frac{1}{N}\sum_{n}\mu_n.\end{aligned} \tag{2}$$

The variance $\sigma_{\mathcal{V}}^2$ for the MoG is derived as follows:

$$\begin{aligned}\sigma_{\mathcal{V}}^2 &= \int z^2 f_{\mathcal{V}}(z)dz - \mu_{\mathcal{V}}^2 \\ &= \frac{1}{N}\sum_{n}\int z^2 f_{c_n}(z)dz - \mu_{\mathcal{V}}^2 \\ &= \frac{1}{N}\sum_{n}(\sigma_n^2+\mu_n^2) - \mu_{\mathcal{V}}^2 \\ &= \frac{1}{N}\sum_{n}(\sigma_n^2+\mu_n^2) - (\frac{1}{N}\sum_{n}\mu_n)^2.\end{aligned} \tag{3}$$

Finally, the geometric mean of the variance $\sigma_{\mathcal{V}}^2$ is used as the uncertainty of $\mathcal{V}$.

B. Probabilistic Distance

B.1. Other Distance Metrics

In the main paper, Bhattacharyya distance is used to measure the probabilistic distance between two normal distributions. We extend the probabilistic distance to various formulations and analyze them in terms of positive and negative mining. To simplify each distance, we first define Euclidean distance[1] as $d(\cdot)$ between two sampled embeddings:

$$d(z_p^{(k)}, z_q^{(k')}) = (z_p^{(k)} - z_q^{(k')})^{\top}(z_p^{(k)} - z_q^{(k')}), \tag{4}$$

where $z_p^{(k)}$ and $z_q^{(k')}$ are sampled from different distributions $p = \mathcal{N}(\mu_p, \sigma_p^2)$ and $q = \mathcal{N}(\mu_q, \sigma_q^2)$, respectively.
Euclidean distance is used to the deterministic counterpart to the probabilistic distance. The distance between two probability distributions are computed via Monte-Carlo estimation:

$$ED(p,q) = \frac{1}{K^2}\sum_{k}\sum_{k'} d(z_p^{(k)}, z_q^{(k')}). \tag{5}$$

Kullback-Leibler (KL) divergence [2] measures the relative entropy of given two probability distributions as fol-

*Corresponding author.

[1]Strictly the square of Euclidean distance.

lows:

$$KL(p,q) = \int \log \frac{p}{q} dp \\ = \frac{1}{2}[\log \frac{\sigma_q^2}{\sigma_p^2} + \frac{\sigma_p^2}{\sigma_q^2} + \frac{ED(p,q)}{\sigma_q^2}]. \quad (6)$$

To enforce a distribution-wise symmetric measure, we employ **Jensen-Shannon (JS) divergence** [6] instead of directly using KL-divergence:

$$JS(p,q) = \frac{1}{2}[KL(p,q) + KL(q,p)]. \quad (7)$$

Wasserstein distance measures the probability distance of two distributions on a given metric space M. The 2-Wasserstein distance between two Gaussian distributions is defined as:

$$W(p,q)^2 = ED(p,q) + \sigma_p - \sigma_q^2. \quad (8)$$

B.2. Comparison on Toy Experiment

Now we compare the mining capacity through the toy experiment on 11 subclasses from UCF101 [11] dataset[2]. We construct a batch by sampling 8 videos per class such that the batch size is 88 and the number of hard positive pairs (having the same class) is 308. Different from the main text, we set threshold distance τ as the average value of self-distances:

$$\tau = \frac{1}{B}\sum_{i=1}^{B} \text{dist}(\mathcal{V}_i, \mathcal{V}_i), \quad (9)$$

where B is batch size and $\text{dist}(\mathcal{V}_i, \mathcal{V}_i)$ represents the average distance between embedding pairs sampled from the video distribution $p(z|\mathcal{V}_i)$. We compute the precision and recall at every 20 epochs for the constructed positive pairs according to the type of distance to estimate the mining capacity, as shown in Fig. 1. At the beginning of training, high recall is shown on the right side of Fig. 1 regardless of the distance due to a large number of positive candidates, while the precision is significantly low, as shown on the left side of Fig. 1. The comparison between Euclidean distance and the probabilistic distances implies that Euclidean distance requires sufficient training steps to eliminate false positives, showing high recall and low precision compared to the probabilistic distance. The results on the probabilistic distance show that Bhattacharyya distance achieves constructing reliable positive pairs with a small number of epochs.

[2]The 11 classes ("ApplyEyeMakeup", "Archery", "BabyCrawling", "BalanceBeam", "BandMarching", "BaseballPitch", "Basketball", "BenchPress", "Biking", "GolfSwing", and "SkyDiving") are used for this experiment.

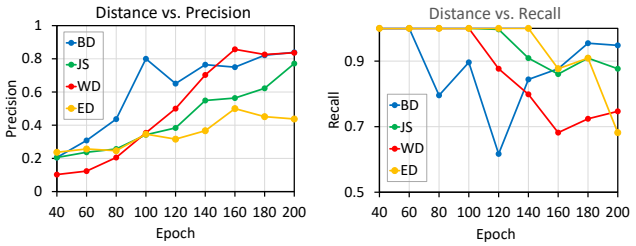


Figure 1. **(Left) Distance versus precision** and **(Right) Distance versus recall** at every 20 epochs. Since we do not consider hard positives at the first 30 epochs, the precision and recall are computed from epoch 40. **BD**: Bhattacharyya distance, **JS**: Jensen-Shannon divergence, **WD**: Wasserstein distance, and **ED**: Elidean distance.

C. Implementation Details

C.1. Two-stage Training

In pretraining, our ProViCo trains the model in a two-stage procedure for stable training following [3, 9]. At the first 30 epochs, the model is trained without hard positive mining such that the positive pairs are defined as

$$\mathcal{P} = \{(\mathcal{V}_i, \mathcal{V}_j) \mid i = j\}. \quad (10)$$

After initial training, the positive and negative pairs are selected based on the probabilistic distance as described in the main text. The two-stage training enables the model to obtain more substantial initial parameters than randomly initialized models and construct the confident training pairs.

C.2. Finetuning and Inference

Finetuning. After self-supervised pretraining, we finetune the model on UCF101 [11] or HMDB51 [5] datasets for action recognition. As same with the pretraining, we randomly sample two 16-frame clips with the temporal stride of 1 from each video and all frames are fixed to a size of 112×112 by random cropping. The backbone network with an additional fully-connected (FC) layer are trained for 200 epochs with a mini-batch size of 96 and the learning rate of 0.02. To predict the action class of a video $\mathcal{V}$, the FC layer takes all embeddings $\{z^{(k)}\}_{k=1}^{K}$ sampled from $p(z|\mathcal{V})$ as inputs and outputs the sample-wise class probabilities $\{y^{(k)}\}_{k=1}^{K}$ followed by a softmax function. We apply cross-entropy loss between the averaged probability score $\frac{1}{K}\sum_k y^{(k)}$ and the groundtruth to learn parameters.
Inference. For action recognition, we uniformly sample 10 clips for each video and apply center cropping for fixed size of 112×112, following [9, 13]. Specifically, the consecutive two clips are used to estimate the video distribution, such that five video distributions are estimated from 10 clips. We sample K embeddings on each distribution and predict $5K$ class probabilities. Therefore, the final prediction of each video is the averaged probability of $5K$ embeddings. For

Number of clips	($\beta = 10^{-4}$, $K = 10$, $B = 40$)			
Parameter N	1	2	3	4
Acc. (%)	78.3	81.8	82.2	**82.4**
Batch size	($\beta = 10^{-4}$, $K = 10$, $N = 2$)			
Parameter B	24	48	72	96
Acc. (%)	79.9	82.1	84.5	**86.1**

Table 1. Ablation studies for the number of clips M and batch size B. We report the action recognition performance evaluated on UCF101 [11] dataset.

Mining	Recognition	Retrieval			
	Acc. (%)	R@1	R@5	R@10	R@20
✗	84.2	61.2	75.5	84.0	88.7
✓	**86.1**	**64.7**	**78.5**	**87.9**	**91.1**

Table 2. Ablation study for positive and negative mining. We evaluate the action recognition and video retrieval performance on the UCF101 [11] dataset.

video retrieval task, we basically follow the experimental protocol in [7, 9] and use the pretrained model without any additional training. Each video in the test split is taken as a query and top-k nearest-neighbors are retrieved from the training set. To this end, the match probability [1, 8] is employed to measure the similarity, and the retrieval performance is evaluated by top-k R@k.

C.3. Code

More details of network architecture and implementation is described in Pytorch-based code supplementary files ('ProViCo.zip'). The whole code will be publicly released after the review process.

D. Additional Results

In this section, we provide additional experimental results with ProViCo. The results include ablation studies, 3D visualization, and qualitative results for video retrieval.

D.1. Ablation Study

We provide additional ablation studies for the number of clips N used during training, batch size B, effectiveness of positive and negative mining, and the similarity metrics.

Number of clips and batch size. We report the action recognition performance evaluated on UCF101 [11] dataset corresponding to the number of input clips N used during training in the first block of Tab. 1. With the limited GPU memory, we set batch size to 40 for all results to eliminate the effect from batch size. Since more sophisticated and complicated distribution for the whole video can be estimated by combining a larger number of clip distributions, the performance is improved as N increases. However, the results that two clips are sufficient to represent the whole video distribution, showing competitive performance to the larger number of clips. In addition, we observe that batch size has a more impact on the performance than the number of clips as shown in the second block of Tab. 1. To consider the trade-off in terms of the computational capacity and the space complexity, we use two clips ($N = 2$) for each video in the main experiments. This also provides fair comparisons with previous methods [4, 10] under general experimental settings.

Similarity Metric	R@1	R@5	R@10	R@20
Cosine Similarity	63.1	77.0	86.3	89.8
Match Probability	**64.7**	**78.5**	**87.9**	**91.1**

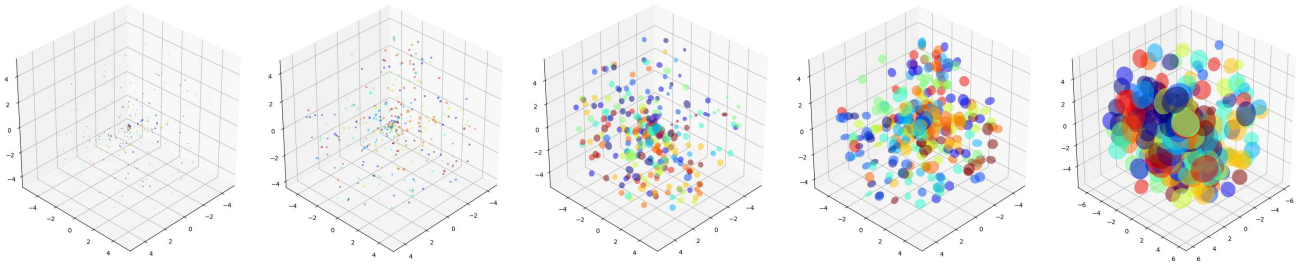
Table 3. Ablation study for the similarity metric on video retrieval. We evaluate video retrieval performance on UCF101 [11] dataset.

Positive and negative mining. We ablate positive and negative mining to verify the effectiveness of the hard positive pairs by evaluating the action recognition and video retrieval performances on UCF101 [11] dataset. As shown in Tab. 2, constructing the positive and negative pairs based on the probabilistic distance obviously improves the performance of downstream tasks, especially on video retrieval.

Similarity metric on video retrieval. We mainly use the match probability [1, 8] as a similarity metric to retrieve videos. To verify the performance on variants of the similarity metric, we evaluate the video retrieval performance in Tab. 3. For the cosine similarity, we use the mean vector of each video to measure the similarity without embedding sampling. The results show that the performance using the match probability is better than the cosine similarity. On the other hand, in terms of space complexity, the cosine similarity and match probability require $\mathcal{O}(N)$ and $\mathcal{O}(K^2N)$ spaces, respectively. Therefore, we can alternatively use the match probability to obtain more accurate retrieval performance and the cosine similarity for the fast inference.

D.2. KL Regularization with 3D Visualization

To visually observe the learned representations and the impact of the KL-divergence hyperparameter β, we conduct an additional toy experiment on 11 subclasses used in Sec. B. Specifically, we slightly transform the architecture of ProViCo to learn 3-dimensional embeddings. We use the same architecture as the main experiments, but add two additional projection layers that take $g_\mu(v_{c_n})$ and $g_\sigma(v_{c_n})$ as an input respectively to obtain 3-dimensional embeddings. Note that we use all the videos from three train splits for 11 classes in this experiment. In Fig. 2, we visualize learned embeddings on 3D space according to β. As analyzed in Sec. 4.5 of the main text, embeddings becomes like points at the small value ($\beta = 10^{-6}$). On the contrary to this, an increase in β leads to an increase in the variance of embeddings, such that embeddings approach to the unit Gaussian at the large value ($\beta = 10^{-2}$).



(a) $\beta = 10^{-6}$ (b) $\beta = 10^{-5}$ (c) $\beta = 10^{-4}$ (d) $\beta = 10^{-3}$ (e) $\beta = 10^{-2}$

Figure 2. **Impact of the KL-divergence hyperparameter β.** We visualize 3-dimensional embeddings learned with the different values of β on 11 subclasses of UCF101 [11] dataset. Each class is viewed in color.

Figure 3. **Qualitative results for video retrieval.** (Left) Input query videos, (Middle) top-3 nearest-neighbors retrieved by the model trained without hard positives, and (Right) top-3 nearest-neighbors retrieved by the model trained with hard positives. The red box indicates the wrong retrieval results.

D.3. Qualitative Results

In Fig. 3, we visualize video retrieval results obtained by top-3 nearest-neighbors on the test split 1 of UCF101 [11] dataset. The results show that the model trained with hard positives more robust to the semantic instance discrimination than the model trained without any hard positives.

E. Extension with Uncertainty

As mentioned in Sec. 1 of the main text, our probabilistic framework can make useful applications such as estimation of difficulty or chance of failure on test data. In this section, we study the uncertainty with corrupted test videos by establishing two factors that increase the uncertainty of the video: (1) the ambiguous content of the video, which is unrelated to the subject of the video, and (2) the empty content of the video, which includes the meaningless background frames.

E.1. Uncertainty on Mixed Clips

To make the content of the video uncertain, we generate clips by mixing frames from several videos. We depict examples of generated mixed clips in Fig. 4. We divide the uncertainty into five levels according to the number of video clips used to generate mixed clips. For example, Fig. 4(c), representing mixture level 3, composes a 16-frame clip by mixing each of four consecutive frames sampled from four videos. We measure the averaged uncertainty of videos from three test splits on UCF101 [11] dataset according to uncertainty levels. As shown on the left side in Fig. 6, the uncertainty increases as the number of mixed videos increases. Based on this result, the uncertainty predicted by our method can be used to eliminate the ambiguous clip or balance the weights between clips for reliable performance on various downstream tasks.

E.2. Uncertainty on Masked Clips

To consider the case where the video contains background frames, we randomly remove frames of the clip and insert random noise frames to removed positions. We divide the background degree according to the removal ratio ρ (*i.e.*, $\rho = 0, 0.2, 0.4, 0.6, 0.8$), as shown in Fig. 5. The results on the right side in Fig. 6 show the uncertainty exponentially increases as the background ratio increases. In practice, *untrimmed raw videos* consist of sparse frames related to the subject of the video and the majority of meaningless background frames. Our probabilistic approach with the uncertainty estimation enables us to effectively exploit the untrimmed videos by filtrating the ambiguous or meaningless contents.

(a) Level 1 (b) Level 2 (c) Level 3 (d) Level 4 (e) Level 5

Figure 4. **Generation of mixed (uncertain) clips.** The uncertainty divided into five levels according to the number of videos used to generate mixed clips.

(a) $\rho = 0$ (b) $\rho = 0.2$ (c) $\rho = 0.4$ (d) $\rho = 0.6$ (e) $\rho = 0.8$

Figure 5. **Generation of masked (background) clips.** The frames in the original clip are replaced by noise frames according to the removal ratio ρ.

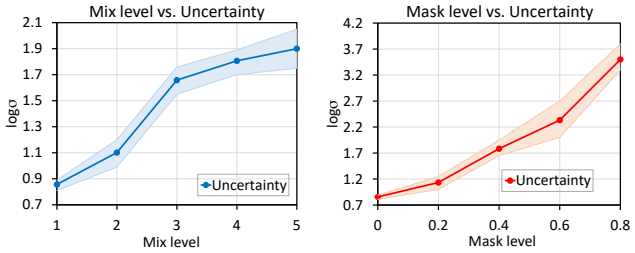


Figure 6. **(Left) Mix level versus uncertainty** and **(Right) Mask level versus uncertainty.** We measure the uncertainty of videos on three test splits of UCF101 [11] dataset according to each corrupted level.

References

[1] Sanghyuk Chun, Seong Joon Oh, Rafael Sampaio de Rezende, Yannis Kalantidis, and Diane Larlus. Probabilistic embeddings for cross-modal retrieval. In *CVPR*, 2021. 3

[2] Thomas M. Cover and Joy A. Thomas. *Elements of information theory*. Wiley, 2006. 1

[3] Tengda Han, Weidi Xie, and Andrew Zisserman. Self-supervised co-training for video representation learning. In *NIPS*, 2020. 2

[4] Simon Jenni and Hailin Jin. Time-equivariant contrastive video representation learning. In *ICCV*, 2021. 3

[5] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. Hmdb: A large video database for human motion recognition. In *ICCV*, 2011. 2

[6] Jianhua Lin. Divergence measures based on the shannon entropy. *IEEE Trans. Inform. Theory*, 37(1):145–151, 1991. 2

[7] Yuanze Lin, Xun Guo, and Yan Lu. Self-supervised video representation learning with meta-contrastive network. In *ICCV*, 2021. 3

[8] Seong Joon Oh, Kevin Murphy, Jiyan Pan, Joseph Roth, Florian Schroff, and Andrew Gallagher. Modeling uncertainty with hedged instance embedding. In *ICLR*, 2019. 3

[9] Rui Qian, Yuxi Li, Huabin Liu, John See, Shuangrui Ding, Xian Liu, Dian Li, and Weiyao Lin. Enhancing self-supervised video representation learning via multi-level feature optimization. In *ICCV*, 2021. 2, 3

[10] Rui Qian, Tianjian Meng, Boqing Gong, and Ming-Hsuan Yang. Spatiotemporal contrastive video representation learning. In *CVPR*, 2021. 3

[11] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from from videos in the wild. *arXiv preprint arXiv:1212.0402*. 2, 3, 4, 5

[12] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *CVPR*, 2018. 1

[13] Jiangliu Wang, Jianbo Jiao, and Yun-Hui Liu. Self-supervised video representation learning by pace prediction. In *ECCV*, 2020. 2