Supplementary Material of Siamese Prototypical Contrastive Learning

Shentong Mo[1]
shentonm@andrew.cmu.edu
Zhun Sun*[2]
sun@vision.is.tohoku.ac.jp
Chao Li[3]
chao.li@riken.jp

[1] Carnegie Mellon University
Pittsburgh, PA 15213, United States
[2] Tohoku University
Sendai, Miyagi, Japan
[3] Center for Advanced Intelligence Project (AIP), RIKEN
Tokyo, Japan

In this supplementary material, we provide additional information on the following topics.

1. Detailed framework in Section 1 for explicit explanation;
2. Detailed experiments configurations in Section 2;
3. More extensive experiments on CIFAR-10, CIFAR-100, ImageNet-100, MS COCO, PASCAL VOC, and Cityscapes to demonstrate the effectiveness of our SPCL in Section 3
4. More visualizations of our SPCL pre-trained features distribution space for explicit interpretation in Section 4;
5. Visualization of cluster quality with different batch size and number of prototypes in Section 5;
6. Visualization of cluster quality with different temperature in Section 6;
7. Influence of symmetric/asymmetric losses in Section 7.

*Corresponding author.

1 Detailed Framework

We provide the main framework of the proposed Siamese Prototypical Contrastive Learning in Figure 1. The first step of our method is to group the embedded features into separate clusters in an unsupervised manner. This is achieved by the simple $k-$means algorithms, where the number of prototypes K is a preset hyper-parameter. Concretely, we warm-up the feature extractor f to obtain the capability to extract features roughly (in this paper, we employ the standard SimCLR framework to complete this). After that, at the beginning of every epoch, we split the dataset $\mathcal{X}$ into mini-batches. For each mini-batch, we draw an augmentation $t \sim \mathcal{T}$ and extract the features of the augmented samples. We concatenate all the features and conduct the clustering algorithm to obtain the set of prototypes $\mathcal{C} = \boldsymbol{c}^1, \ldots, \boldsymbol{c}^K$, where $\boldsymbol{c}$ is the vector of index recording samples of the cluster c.

Next, we draw an anchor prototype p and sample a mini-batch $\mathcal{X}^p = \{\boldsymbol{x}_n^p\}_{n=1}^N$ from $\boldsymbol{c}^p$; Then we draw samples that do not belong to prototype p, *i.e.* from $\mathcal{C} \setminus \{\boldsymbol{c}^p\}$. With slight abuse of notation, we denote this mini-batch with $\mathcal{X}^q = \{\boldsymbol{x}_n^q\}_{n=1}^N$, where q stands for any prototypes that are not p. For each sample in $\boldsymbol{x}^p, \boldsymbol{x}^q$, we acquire its *two* augmented views $\tilde{\boldsymbol{x}}_i^c, \tilde{\boldsymbol{x}}_j^c$ and the feature representations $\boldsymbol{h}_i^c, \boldsymbol{h}_j^c$ in the embedding space. After that, we apply a Siamese-style metric loss to minimize the agreement between the intra-prototype feature representations, and maximize the agreement between the inter-prototype representations. For the contrastive loss, we use a small neural network projection head $g_c(\cdot)$ that maps the extracted representations $\boldsymbol{h}$ to the embeddings $\boldsymbol{z}$.

In order to guide the views of one sample to its corresponding prototype, we introduce the prototypical cross-entropy framework, as shown in Figure 2. Specifically, we use a linear projection head $g_p(\cdot)$ that maps the representations to the label, *i.e.* the corresponding prototype (label) of the sample. The prototypical cross entropy loss term is helpful when the similarity between views is small, *i.e.* $s(\boldsymbol{z}, \boldsymbol{z}') = 0$, and the contrastive loss term only produce gradients with trivial magnitude.

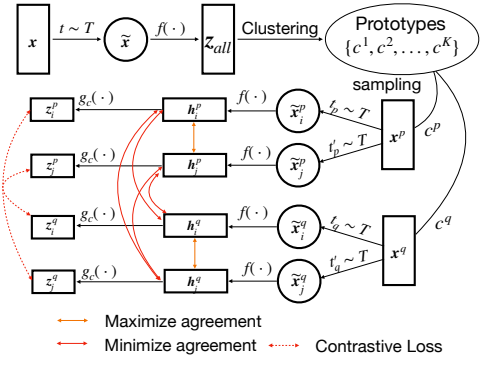

Figure 1: Main framework for SPCL.

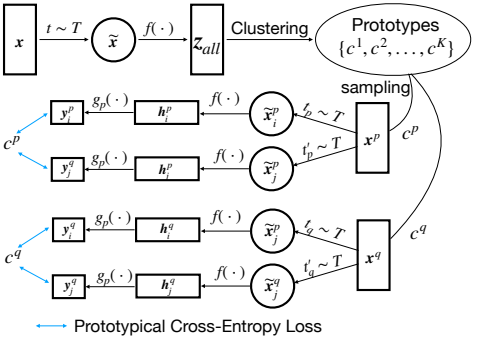

Figure 2: Prototypical cross-entropy framework.

2 Detailed Experiments Configurations

In this section, we provide the detailed experiments configurations of pre-training and downstream tasks setup on various benchmarks.

CIFAR 10/100. For CIFAR-10/100 [14] pre-training, we closely follow SimCLR [2] and use the same data augmentation, *i.e.*, random crop and resize (with random flip), color distortions, and Gaussian blur. For the encoder network ($f(\cdot)$) we experiment with three commonly used encoder architectures, *i.e.*, ResNet-18, ResNet-50, and ResNet-101. As the optimizer, we use LARS [24] with learning rate of 1.0 and weight decay of 10^{-6}. We use linear warmup for the first 10 epochs, and decay the learning rate with the cosine decay schedule. We train at batch size 512 for 1000 epochs, using 512 prototypes for clustering. For clustering, we adopt the faiss library [11] for efficient k-means clustering, where it takes 1.5 seconds per epoch during the pre-training. The whole training time is 10 hours using 8 Tesla V100 GPUs. After pre-training, we implement the linear evaluation on the pre-trained representations.

ImageNet-1K. For ImageNet-1K [7] of 0.32 million total images pre-training, following [4, 21], we train it for 200 epochs. We use LARS with an initial learning rate of 1.0 and weight decay of 10^{-6} with the cosine decay schedule. We train at a batch size of 1024, using 2048 prototypes in our clustering stage. The clustering only takes 6.5 seconds per epoch. The clustering only takes less than 1300 seconds in total, which is fairly negligible compared to the whole training time (482 hours using 8 Tesla V100 GPUs).

ImageNet-100. For the ImageNet-100 pre-training, we use a subset of ImageNet-1K that contains randomly selected 100 classes following CMC [18], MoCHI [12]. The rest of the settings (including optimizer, weight decay, etc.) are the same as our ImageNet-1K training. The clustering only takes 3.2 seconds per epoch. The clustering only takes less than 640 seconds in total, which is fairly negligible compared to the whole training time (145 hours using 8 Tesla V100 GPUs).

MS COCO. For COCO of 118k images totally, we use an initial learning rate of 0.3. We adopt SGD as the optimizer and we set its weight decay and momentum to 0.0001 and 0.9. Both pre-training models are optimized with a cosine learning rate decay schedule and a mini-batch size of 1024. We use 2048 prototypes during the clustering stage. We train 800 epochs for COCO, which is a total of 92k iterations. The clustering only takes 2.5 seconds per epoch. The clustering only takes less than 2000 seconds in total, which is fairly negligible compared to the whole training time (234 hours using 8 Tesla V100 GPUs). After pre-training, we evaluate object detection and instance segmentation by fine-tuning a Mask R-CNN detector (FPN-backbone) in Detectron 2 on COCO `train2017` split with standard $1\times$ schedule and evaluating on COCO 5k `val2017` split.

PASCAL VOC. Following DenseCL [21], we use MS COCO [23] and ImageNet-1K [7] for separate pre-training. Then, we evaluate object detection by fine-tuning a Faster R-CNN detector (C4-backbone) on the VOC `trainval07+12` [8] set with standard 2x schedule in Detectron 2 and testing on the VOC `test2007` set. We also evaluate semantic segmentation by fine-tuning an FCN model [16] on VOC `train_aug2012` set for 20k iterations and testing on `val2012` set.

Cityscapes. For pre-training, we closely follow [21], and perform the pre-training on COCO and ImageNet-1K, separately. We apply the same pre-training setting as we set in the pre-training for object detection. We evaluate semantic segmentation on Cityscapes [6] dataset by training an FCN model on `train_fine` set for 40k iterations and test on `val` set.

3 More Experiment Results

CIFAR-10. In Table 1, we report the top-1/top-5 accuracy of image classification on CIFAR-10 and compare it with SimCLR [2] using three backbones (ResNet-18, ResNet-50, ResNet-101). The results are averaged over 5 independent trials.

Table 1: Top-1 and top-5 accuracy on the CIFAR-10 dataset.

Method	Arch.	top-1(%)	top-5(%)
SimCLR	ResNet-18	89.80	99.75
SPCL	ResNet-18	**94.12**(↑ 4.32)	**99.80**(↑ 0.05)
SupCon	ResNet-18	94.90	-
SimCLR	ResNet-50	92.00	99.81
SPCL	ResNet-50	**94.42**(↑ 2.42)	**99.86**(↑ 0.05)
SupCon	ResNet-50	96.00	-
SimCLR	ResNet-101	93.06	99.85
SPCL	ResNet-101	**95.32**(↑ 2.26)	**99.90**(↑ 0.05)

Table 2: Top-1 and top-5 accuracy on the CIFAR-100 dataset.

Method	Arch.	top-1(%)	top-5(%)
SimCLR	ResNet-18	57.26	85.69
SPCL	ResNet-18	**69.69**(↑ 12.43)	**92.24**(↑ 6.55)
SupCon	ResNet-18	71.90	-
SimCLR	ResNet-50	61.58	88.11
SPCL	ResNet-50	**73.97**(↑ 12.39)	**93.24**(↑ 5.13)
SupCon	ResNet-18	76.50	-
SimCLR	ResNet-101	62.98	88.79
SPCL	ResNet-101	**75.36**(↑ 12.38)	**93.38**(↑ 4.59)

CIFAR-100. Similar to CIFAR-10, the top-1 and top-5 accuracy results using three backbones (ResNet-18, ResNet-50, ResNet-101) on CIFAR-100 are reported in Table 2. The results are averaged over 5 independent runs.

Compared to SupCon [13], our SPCL achieves comparable results on CIFAR-10 and CIFAR-100 in terms of top-1 accuracy. This further shows the effectiveness of our SPCL, although we did not use the ground-truth as the supervision.

Table 3: Top-1 and top-5 accuracy on the ImageNet-100 dataset. Bold and underline denote the first and second place.

Method	Arch.	top-1(%)	top-5(%)
MoCo[11]	ResNet-50	72.80	91.64
Biased CMC [18]	ResNet-50	73.58	92.06
Debiased CMC [3]	ResNet-50	<u>74.60</u>	92.08
MoCo+align/uniform[20]	ResNet-50	<u>74.60</u>	<u>92.74</u>
SPCL (ours)	ResNet-50	**74.72**(↑ 0.12)	**92.78**(↑ 0.04)

ImageNet-100. We also report the performance on ImageNet-100 which has be adopted by former studies, the classes in this sub-set are chosen as [5, 20]. The results are reported in Table 3, our SPCL outperforms MOCO by 1.92%, 1.14% in terms of top-1 and top-5 accuracy. We also achieve better result than [5] and [20], which further demonstrates the advantage of our SPCL on achieving the uniformity in the prototype-level instead of instance-level.

Object detection. In Table 4, we report the object detection result on PASCAL VOC [8] and COCO. The metrics include the VOC metric AP_{50} and COCO-style AP and AP_{75}. The results are averaged over 5 independent trials. As can be seen, the COCO pre-trained SPCL achieves the new state-of-the-art in terms of AP_{75} and AP_{50} compared with MoCov2 and DenseCL. The ImageNet pre-trained SPCL outperforms MoCov2 by a large margin, 1.55% and 1.71%, in terms of AP and AP_{75}. When pre-trained on ImageNet, our SPCL achieves comparable performance with DenseCL [21] and SCRL [17]. These two methods are particularly designed and focusing on pixel-level and spatial-level contrastive learning, respectively. For evaluation on COCO, our ImageNet pre-trained SPCL outperforms the MoCov2 by 0.85%, 0.22% and 0.90%. The COCO pre-trained SPCL achieves new best results in terms of AP^b_{75} and AP^b_{50} pre-trained on COCO. These results shows the advantage of our SPCL as a general proposal pre-training method, and the capacity of transferring learned representations to different vision tasks.

Instance Segmentation. In Table 4, we also report the results on COCO for evaluation. When pre-trained on both ImageNet and COCO, our SPCL outperforms the PCL [15] and DenseCL [21] in terms of AP^m_{50}. We also achieve competitive results compared with [9, 21] in terms of AP^m and AP^m_{75} on the instance segmentation downstream task.

Semantion Segmentation. The semantic segmentation results on the PASCAL VOC and Cityscapes are reported in Table 4. When pre-trained on ImageNet, our SPCL outperforms MoCov2 by a large margin, i.e., 1.85% mIoU. We are comparable to the expert method DenseCL, although we do not apply level of pixels by taking into account the correspondence between local features. The COCO pre-trained SPCL is competitive with the DenseCL and outperforms the MoCov2 by 3.05% mIoU. For evaluation on Cityscapes, we observe the same performance boost with our SPCL compared with MoCov2, i.e., 1.22% pre-trained on ImageNet, and 1.85% on COCO. Our SPCL also achieves competitive performance.

Table 4: Comparison results of Object Detection (**OD**) fine-tuned on PASCAL VOC/COCO, Instance Segmentation (**IS**) on COCO, and Semantic Segmentation (**SS**) on VOC/Cityscapes.

	VOC (OD)			COCO (OD)			COCO (IS)			VOC (SS)	City (SS)
Method	AP	AP_{50}	AP_{75}	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}	$mIoU_{VOC}$	$mIoU_{City}$
Random Initialization	32.80	59.00	31.60	32.80	50.90	35.30	29.90	47.90	32.00	40.70	63.50
Models pre-trained using ImageNet-1K:											
Supervised (CE)	54.20	81.60	59.80	39.70	59.50	43.30	35.90	56.60	38.60	67.70	73.70
SimCLR [6]	51.50	79.40	55.60	—	—	—	—	—	—	—	—
BOYL [9]	51.90	81.00	56.50	—	—	—	—	—	—	—	—
SwAV [5]	55.40	81.50	61.40	37.60	57.60	40.30	33.10	54.20	35.10	—	—
MoCo [21]	55.90	81.50	62.60	40.70	60.50	44.10	35.40	57.30	37.60	—	—
MoCov2 [4]	57.00	82.40	63.60	39.80	59.80	43.60	36.10	56.90	38.70	67.50	74.50
MoCHi [13]	57.50	82.70	64.40	39.40	59.00	42.70	34.50	55.70	36.70	—	—
SimSiam [8]	57.00	82.40	63.70	39.20	59.30	42.10	34.40	56.00	36.70	—	—
SCRL [17]	57.20	**83.80**	63.90	—	—	—	—	—	—	—	—
PCL [15]	—	—	—	**41.00**	**60.80**	44.20	35.60	57.40	37.80	—	—
DenseCL [21]	**58.70**	82.80	65.20	40.30	59.90	44.30	**36.40**	57.00	**39.20**	**69.40**	75.70
SPCL(ours)	58.55	82.63	**65.31**	40.65	60.02	**44.50**	36.21	**57.41**	38.57	69.35	**75.72**
Models pre-trained using COCO:											
MoCov2 [4]	54.70	81.00	60.60	38.50	58.10	42.10	34.80	55.30	37.30	64.50	73.80
DenseCL [21]	**56.70**	81.70	63.00	**39.60**	59.30	43.30	**35.70**	56.50	**38.40**	67.50	75.60
SPCL(ours)	56.54	**81.76**	**63.15**	39.54	**59.45**	**43.42**	35.65	**56.89**	38.31	**67.55**	**75.65**

4 Visualization of Feature Distributions

We provide more visualization of feature distributions under different datasets and conditions. Specifically, we implement the ResNet-50 pre-trained models on the 10 classes from

CIFAR-10 dataset, 20, 100 classes chosen from the CIFAR-100 dataset. Then we project the pre-trained representations of 2048-dimension onto the 2-dimension space using the t-SNE [19], as shown in Figure 3, 4, and 5. We can observe that the SPCL pre-trained representations are distributed more uniformly on the space in terms of 10, 20, 100 classes. This means that each cluster is clustered more inside, and different clusters are scattered more globally in the embedding space.

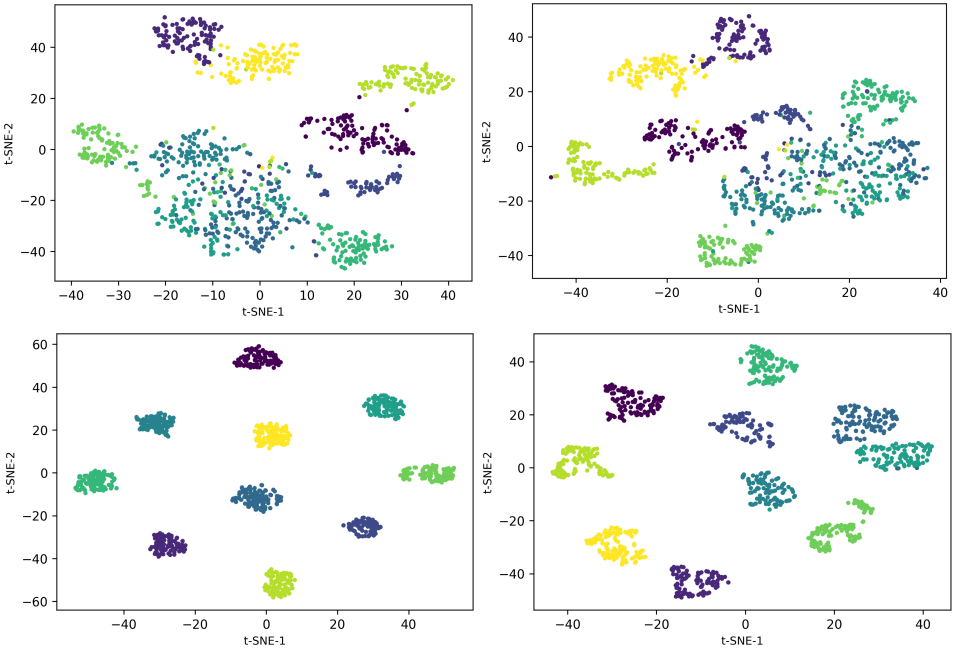

Figure 3: Visualization of the SimCLR (**Top Row**) and SPCL (**Bottom Row**, ours) pre-trained representations (**Left Column**) and features (**Right Column**) after projection head from *10* classes in the CIFAR-10 dataset.

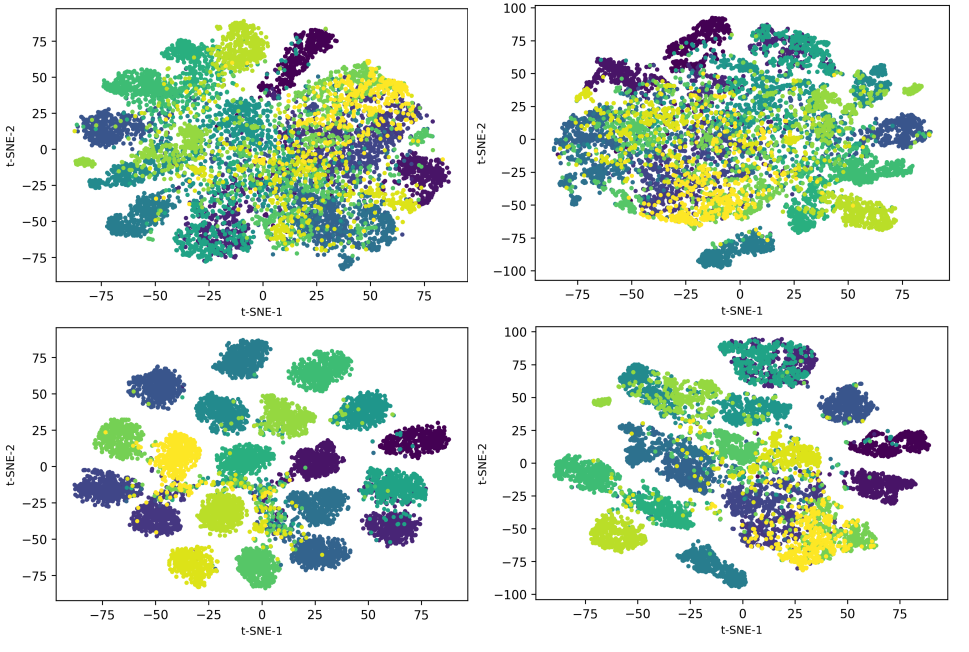

Figure 4: Visualization of the SimCLR (**Top Row**) and SPCL (**Bottom Row**, ours) pre-trained representations (**Left Column**) and features (**Right Column**) after projection head from *20* classes in the CIFAR-100 dataset.

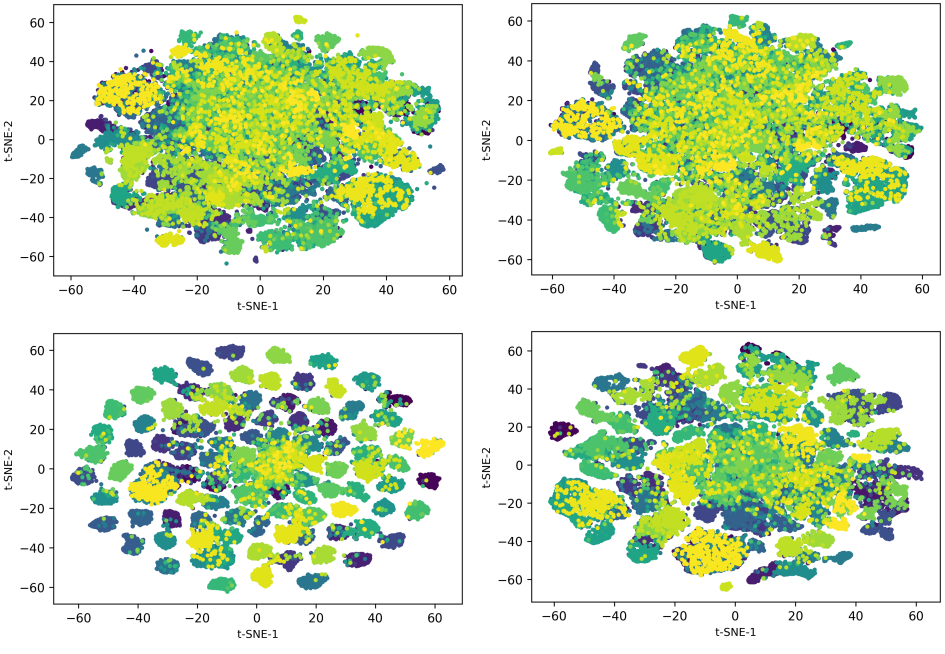

Figure 5: Visualization of the SimCLR (**Top Row**) and SPCL (**Bottom Row**, ours) pre-trained representations (**Left Column**) and features (**Right Column**) after projection head from *100* classes in the CIFAR-100 dataset.

5 Visualization of Cluster Quality with Different Batch size and Number of Prototypes

In this section, we further evaluate the quality of the pre-trained feature distributions, using different batch sizes and number of prototypes in Figure 6. As can be seen, with the prototypes of 512, our SPCL pre-trained representations are indeed more centralized inside each separate cluster compared to the prototype of 256. This infers that the prototypes are indeed crucial for our SPCL to help learn meaningful representations. From the visualization of feature distributions using batch sizes of $256, 512, 1024$, and 2048, we can observe an insignificant change of the quality of learned representations. This further shows the robustness of our SPCL to the choice of the batch size.

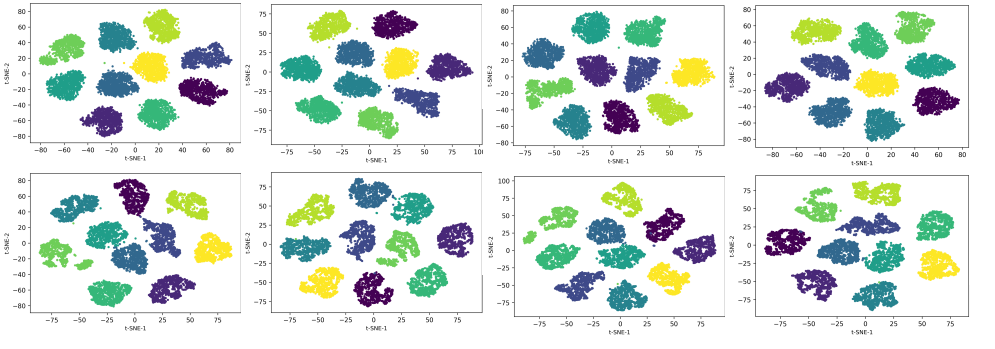

Figure 6: Visualization of the SPCL pre-trained representations using 256 (**Top Row**) and 512 (**Bottom Row**) prototypes and 256 (**First Column**), 512 (**Second Column**), 1024 (**Third Column**), and 2048 (**Last Column**) batch sizes for *10000* samples in CIFAR-10 test set.

6 Visualization of Cluster Quality with Different Temperature

In addition, we evaluate the quality of the pre-trained feature distributions for different temperatures during our SPCL pre-training stage in Figure 7. As the temperature τ increases, it gives a larger penalty to the samples closed to the anchor, which leads to the increase of the TP distance and the decrease of the FN distance. This leads to our SPCL pre-trained representations more concentrated and uniformly distributed in the feature space. When τ is too small, there is little penalty to the samples closed to the anchor, which would cause two clusters coincide in the space due to the increase of the FN distance.

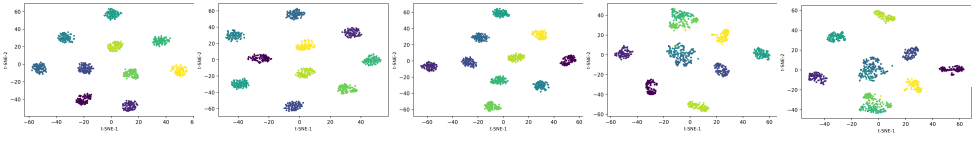

Figure 7: Visualization of the SPCL pre-trained representations using different temperatures, from left to right: $\tau = 0.01$ (**First Plot**), $\tau = 0.05$ (**Second Plot**), $\tau = 0.1$ (**Third Plot**), $\tau = 0.5$ (**Fourth Plot**), and $\tau = 1$ (**Last Plot**) for *1000* samples in CIFAR-10 test set.

7 Influence of Symmetric/Asymmetric Losses

Intuitively, as long as the the number of prototypes does not match the number of classes in the down-stream task, noise in the semantic structure will be introduced. In order to render this phoneme, we implement a symmetric prototypical cross-entropy loss, defined in [22], to overcome the noisy prototypes. We conduct experiments on CIFAR-100 using 512 prototypes, and the symmetric loss achieves better performance (74.04%, 93.35%) compared to asymmetric loss (73.97%, 93.24%) in terms of top-1 and top-5 accuracy. In order to interpret the influence of symmetric and asymmetric losses, we also evaluate the quality of the pre-trained feature distributions when applying symmetric and asymmetric losses during our SPCL pre-training stage in Figure 8. As can be seen in the CIFAR-10 test set (**First/Second Plots**), with the introduction of symmetric loss in our prototypical cross entropy loss module, the pre-trained representations are indeed more centralized inside each separate cluster. We can observe the similar improvement in terms of the quality of our SPCL pre-trained feature distributions in the CIFAR-100 test set on the **Third/Last Plots** in Figure 8.

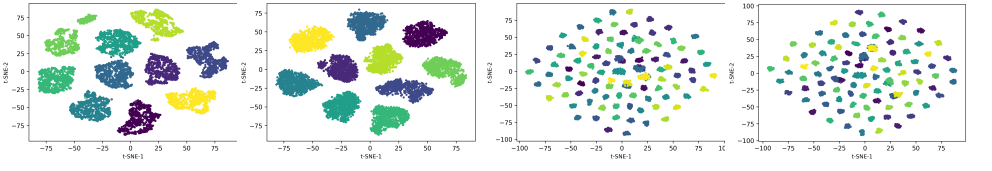

Figure 8: Visualization of the SPCL pre-trained representations using asymmetric (**First/Third Plots**) and symmetric (**Second/Last Plots**) prototypical cross entropy loss in CIFAR-10 (**First/Second Plots**) and CIFAR-100 (**Third/Last Plots**) test set.

References

[1] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[2] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of International Conference on Machine Learning*, 2020.

[3] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15750–15758, 2021.

[4] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

[5] Ching-Yao Chuang, Joshua Robinson, Yen-Chen Lin, Antonio Torralba, and Stefanie Jegelka. Debiased contrastive learning. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[6] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3213–3223, 2016.

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia. Li, Kai Li, and Li Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009.

[8] Mark Everingham, Luc Van Gool, Christopher K. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, pages 303–338, 2010.

[9] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, koray kavukcuoglu, Remi Munos, and Michal Valko. Bootstrap your own latent - a new approach to self-supervised learning. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[10] 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 (CVPR)*, pages 9729–9738, 2020.

[11] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*, 2017.

[12] Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. Hard negative mixing for contrastive learning. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[13] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[14] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

[15] Junnan Li, Pan Zhou, Caiming Xiong, and Steven Hoi. Prototypical contrastive learning of unsupervised representations. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2021.

[16] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440, 2015.

[17] Byungseok Roh, Wuhyun Shin, Ildoo Kim, and Sungwoong Kim. Spatially consistent representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1144–1153, 2021.

[18] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2020.

[19] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[20] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *Proceedings of International Conference on Machine Learning (ICML)*, 2020.

[21] Xinlong Wang, Rufeng Zhang, Chunhua Shen, Tao Kong, and Lei Li. Dense contrastive learning for self-supervised visual pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3024–3033, 2021.

[22] Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. Symmetric cross entropy for robust learning with noisy labels. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 322–330, 2019.

[23] Tsung yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 740–755, 2014.

[24] Yang You, Igor Gitman, and Boris Ginsburg. Scaling SGD batch size to 32k for imagenet training. *arXiv preprint arXiv:1708.03888*, 2017.