ProtoSEED: Prototypical Self-Supervised Representation Distillation

Kyungmin Lee
Agency for Defense Development
kyungmnlee@add.re.kr

Abstract

We propose prototypical self-supervised representation distillation, where it transfers the representational knowledge of a self-supervised network to another one by distilling the prototypical probability distribution. To that end, we present prototypical contrastive predictive coding that combines prototypical methods and contrastive learning by modeling the critic as the probabilistic discrepancy between probabilistic outputs of teacher and student. We demonstrate that the proposed approach dramatically improves the performance of representation learning on small networks and outperforms pre-existing self-supervised model compression baseline.

1 Introduction

Self-supervised learning (SSL) has drawn massive attention due to its aptitude to learn useful features that have outstanding performance on downstream tasks without manual annotation. However, most self-supervised methods involve large networks (such as ResNet-50) and do not work well on small networks. Therefore, [1] proposed self-supervised representation distillation (SEED) that transfers the representational knowledge of a big self-supervised network to a smaller one to aid the representation learning on a small networks.

The contrastive objectives such as CPC [2] or as known as InfoNCE are proven to be effective in learning representations [3, 4, 5]. Especially, CPC is proven to be a lower bound to the mutual information, allowing networks to capture the correlation of representations without a label. However, it requires large negative samples to achieve adequate performance. On the other hand, the prototypical method has shown empirical success in SSL without pertaining memory buffer [6, 7, 8]. They involve prototypes to generate probabilistic outputs and minimize cross-entropy loss between the prototypical distributions, which is adapted from the knowledge distillation (KD) [9].

In this paper, we improve SEED by using prototypical methods. Our contributions are two-fold: First, we propose *prototypical contrastive predictive coding* (ProtoCPC), where we combine contrastive learning and prototypical method by modeling the critic as the probabilistic discrepancy of two prototypical distributions. Then we present *prototypical self-supervised representation distillation* (ProtoSEED), where we use prototypes to transfer the representational knowledge of a large self-supervised model to a smaller one. Our method applies to any self-supervised teacher model and is simple in its implementation. Through experiments on ImageNet, we show that our approach drastically improves the representation learning of ResNet-18. Moreover, we show that our method outperforms SEED even with a shorter epoch of training.

35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.

2 Method

2.1 Prototypical Contrastive Predictive Coding

Let $x \sim X$ be a data with random variable X. Given a teacher network f^T and a student network f^S, which maps x into $\mathbb{R}^{D_T}$ and $\mathbb{R}^{D_S}$ respectively. Let $T = f^T(X)$ and $S = f^S(X)$ be random variables for each teacher and student representations. The contrastive predictive coding (CPC) [2] is a variational lower bound to the mutual information $I(T;S)$ where it is defined by KL-divergence between the joint distribution $p(T,S)$ and the product of the marginal distributiosn $p(T)p(S)$. Formally, given a student feature $z_s \sim S$ with a positive $z_t \equiv z_{t0}$ and $N-1$ negatives $\{z_{tj}\}_{j=1}^{N-1}$ sampled from T, i.e. $(z_t, z_s) \sim p(T,S)$ and $\{(z_{tj}, z_s)\}_{j=1}^{N-1} \sim p(T)p(S)$, the following inequality holds for any critic $h : \mathbb{R}^{D_T} \times \mathbb{R}^{D_S} \to \mathbb{R}_+$:

$$I(T;S) \geq \mathbb{E}\left[\log \frac{h(z_t, z_s)}{\frac{1}{N}\sum_{j=0}^{N-1} h(z_{tj}, z_s)}\right] \tag{1}$$

Usually the critic h is set by the exponential of inner product of two unit feature vectors. On the other hand, we model the critic by the discrepancy of two probability distributions. We use cross-entropy to measure the discrepancy, but since cross-entropy requires two distributions to be on same probability space, we append additional prototypes to match the dimension. Let $W^T \in \mathbb{R}^{D_T \times K}, W^S \in \mathbb{R}^{D_S \times K}$ be a linear prototypical layer which maps each teacher and student features to dimension of K. For probability p_s of a student representation, we use softmax operator on $\tilde{z}_s$ with temperature $\tau_s > 0$:

$$p_s^{(k)} = \frac{\exp\left(\tilde{z}_s^{(k)}/\tau_s\right)}{\sum_{k'=1}^{K} \exp\left(\tilde{z}_s^{(k')}/\tau_s\right)}. \tag{2}$$

Similarly we set probability p_{tj} of teacher representations with temperature $\tau_t > 0$. Then we set the critic by $h(\tilde{z}_t, \tilde{z}_s) = e^{-H(p_t, p_s)} = e^{\sum_{k=1}^{K} p_t^{(k)} \log p_s^{(k)}}$. Remark that the critic is bounded and achieves maximum when p_t is equal to p_s. Then by substitution into Eq. 1, it follows that

$$\begin{aligned} I(T;S) &\geq \mathbb{E}\left[\log \frac{e^{-H(p_t,p_s)}}{\frac{1}{N}\sum_{j=0}^{N-1} e^{-H(p_{tj},p_s)}}\right] = \mathbb{E}\left[\log \frac{\exp(p_t \cdot \tilde{z}_s/\tau_s)}{\frac{1}{N}\sum_{j=0}^{N-1} \exp\left(p_{tj} \cdot \tilde{z}_s/\tau_s\right)}\right] \\ &\geq \mathbb{E}\left[\log \frac{\exp(p_t \cdot \tilde{z}_s/\tau_s)}{\frac{1}{N}\sum_{j=0}^{N-1}\sum_{k=1}^{K} p_{tj}^{(k)} \exp\left(\tilde{z}_s^{(k)}/\tau_s\right)}\right] = \mathbb{E}\left[\log \frac{\exp(p_t \cdot \tilde{z}_s/\tau_s)}{\sum_{k=1}^{K} q^{(k)} \exp\left(\tilde{z}_s^{(k)}/\tau_s\right)}\right], \end{aligned} \tag{3}$$

where $q^{(k)} = \frac{1}{N}\sum_{j=0}^{N-1} p_{tj}$ is a mean of teachers' probability which we refer to *prior*. The first inequality is by data processing inequality. The second equality is by crossing out the constant term C, and third inequality is from Jensen's inequality. We define **Prototypical Contrastive Predictive Coding (ProtoCPC)** objective by last term in Eq. 3. In addition, we define **ProtoCPC loss** $\mathcal{L}_{\text{ProtoCPC}}$ by the negative of ProtoCPC objective, thus minimizing ProtoCPC loss is equivalent to a variational maximization of mutual information between student and teacher representations.

Prior momentum Since ProtoCPC is contrastive, it requires sufficient negatives to perform learning. However, unlike CPC, ProtoCPC only requires prior q that accounts for the negatives. Therefore, we use exponential moving average (EMA) on prior q for better estimation as following:

$$q^{(k)} \leftarrow m_p q^{(k)} + (1 - m_p)\frac{1}{N}\sum_{j=1}^{N} p_{tj}^{(k)}, \tag{4}$$

where $m_p > 0$ is a momentum rate.

Assignment of teacher probability While KD used softmax operator for both probabilities of teacher and student networks, many self-supervised methods [6, 7] reported that the softmax operator can lead to collapse, i.e. every representation fall into the same one. To compromise, many prototypical self-supervised methods resort on *Sinkhorn-Knopp* iterative algorithm by formulating the assignment of teacher probability as an optimal transport problem:

$$\underset{P_t}{\text{argmax}} \ \langle P_t, Z_t \rangle + \tau_t H(P_t), \quad \text{s.t.} \quad P_t \in \mathbb{R}_+^{N \times K}, P_t \mathbf{1}_K = \mathbf{1}_N, P_t^\top \mathbf{1}_N = \frac{N}{K}\mathbf{1}_K, \tag{5}$$

where Z_t is a matrix whose rows are z_{tj}, $H(P_t) = \sum_{j=0}^{N-1} \sum_{k=1}^{K} -p_t^{(k)} \log p_t^{(k)}$ is a entropy and $\tau_t > 0$ is a temperature that controls the smoothness of distribution. Then the Eq. 5 can be solved by only few steps of Sinkhorn-Knopp iteration [10, 11] which iteratively projects P_t into following form:

$$p_{tj}^{(k)} = \frac{\beta_k e^{z_{tj}^{(k)}/\tau_t}}{\sum_{k'=1}^{K} \beta_{k'} e^{z_{tj}^{(k')}/\tau_t}}, \tag{6}$$

where β_k is a normalizing constant. We refer this to Sinkhorn-Knopp (SK) operator.

2.2 Prototypical Self-supervised Representation Distillation

We propose **prototypical self-supervised representation distillation (ProtoSEED)** which uses prototypes to generate the probabilistic output of teacher and student representations, and use ProtoCPC loss for effective distillation. Let the pre-trained teacher network g^T be a composition of base encoder f^T and the projection head h^T. Then we train student network g^S, where it is composed of smaller encoder f^S and projection head h^S of the same architecture as h^T. Then we append prototypes W^T and W^S for each g^T and g^S to ensure that they have the same output of dimension K. Given a data x, we train the student network by minimizing the ProtoCPC loss between the probability of teacher $p_t(x)$ and probability of student $p_s(x)$. For p_s we use softmax operator with temperature $\tau_s > 0$ and for p_t we use SK operator with temperature $\tau_t > 0$ over the batch of samples. Then the objective is given by following:

$$\min_{g^S, W^S} \mathbb{E}_{x \sim X} \left[\mathcal{L}_{\text{ProtoCPC}}(p_t(x), p_s(x)) \right]. \tag{7}$$

For prototypes of teacher network W^T, we copy the parameters of W^S to W^T at each iteration. This allows our method to apply to any self-supervised teacher networks. Note that if the teacher network is trained by prototypical methods such as DINO [8] or SwAV [7], we can re-use the pre-trained prototypes for W^T. Later, we present ablation studies on setting prototypes for W^T.

3 Experiment

Setup We experiment distillation of various self-supervised networks to ResNet-18 [12] on ImageNet [13] without class labels. We consider following self-supervised teacher networks: MoCo-v2 [4] pre-trained ResNet-50, SwAV [7] pre-trained ResNet-50 and DINO [8] pre-trained ResNet-50 and vision transformer [14]. We train for 100 epochs and we additionally conduct experiments on using multi-crops data augmentation for SwAV and DINO pre-trained ResNet-50 networks. For evaluation, we follow linear evaluation protocol which conducts supervised learning on the linear layer appended at the top of the frozen feature and k-nearest neighbor classification (k-NN).

Main results Table 1 show the main results of our self-supervised model compression compared to self-supervised learning (SSL) of itself. We observe that ProtoCPC outperforms SSL with a large margin, especially showing superior performance in the k-NN classification. Note that our method works well for various self-supervised teacher networks and works well across the different architectures of teacher and student (vision transformer teacher to ResNet student).

	MoCo ResNet-50		SwAV ResNet-50		DINO ResNet-50		DINO DeiT-S/16	
	Linear	*k-NN*	*Linear*	*k-NN*	*Linear*	*k-NN*	*Linear*	*k-NN*
Teacher	71.1	61.9	75.3	65.7	75.3	67.5	77.0	74.3
Supervised	69.5	69.5	69.5	69.5	69.5	69.5	69.5	69.5
SSL	52.5	36.7	57.5	48.2	58.2	50.3	58.2	50.3
ProtoSEED	**61.1**(+8.6)	**55.6**(+18.9)	**63.1**(+5.6)	**57.7**(+9.4)	**63.9**(+5.7)	**60.3**(+10.0)	**65.5**(+7.3)	**63.2**(+12.9)

Table 1: Main result of our ProtoSEED on distillation of various self-supervised teacher models to ResNet-18. The teacher models are MoCo ResNet-50, SwAV ResNet-50 and DINO ResNet-50 and DeiT small with patch size 16. The self-supervised denotes the result of self-supervised learning on ResNet-18 with the same method of teacher network.

Comparison with SEED Table 2 compares our method with SEED [1], an original method for self-supervised model compression. We observe that our method consistently outperforms SEED in both k-*NN* and *linear* evaluation with the same teacher network applied. Although training for shorter epochs, our method achieves outperforms SEED. When the teacher is pre-trained by SwAV, our method outperforms SEED without using multi-crops data augmentation, and the gap becomes larger when we use multi-crops as well. When using DINO self-supervised teacher and multi-crops data augmentation, we achieve the best results in representation learning of ResNet-18.

Teacher	Method	Epochs	*Linear*	k-*NN*
MoCo	SEED	200	60.5	49.1
	ProtoSEED	100	**61.1**	**55.6**
SwAV	SEED	100	61.1	-
	SEED	200*	62.6	-
	ProtoSEED	100	63.1	**57.7**
	ProtoSEED	100*	**63.9**	57.0
DINO	ProtoSEED	100	63.9	60.3
	ProtoSEED	100*	**65.3**	**60.7**

Table 2: Comparison of our method with SEED. * denotes training with multi-crops. Every teacher networks are ResNet-50 and student networks are ResNet-18.

Cross-entropy v.s. ProtoCPC and Softmax v.s. Sinkhorn-Knopp In Table 3, we provide ablation on the loss function and assignment method for p_t. We compare ProtoCPC and cross-entropy loss, where cross-entropy is a wide-ranging choice for knowledge distillation. Also, we compare softmax and Sinkhorn-Knopp operator. We observe that using SK operator achieves better performance than softmax operator, and ProtoCPC outperforms cross-entropy loss.

Using teacher's old prototypes In Table 4, we provide ablation on using teacher's old prototypes. We observe that when distilled from SwAV ResNet-50 teacher, using old (pre-trained) prototypes outperform update with student's prototypes. On the other hand, when distilled from DINO ResNet-50, copying from student's prototypes performs better. We suspect that as SwAV used single prototypes throughout pre-training, the pre-trained prototypes contain representational knowledge. On the other hand, the pre-trained prototypes of DINO lack such representational knowledge.

Prior momentum Our ProtoCPC objective relies on the prior term for contrastive learning. Then using momentum for the prior term allows us a more accurate estimation. Table 5 shows the result of ProtoSEED from DINO teachers with varying prior momentum. We observe that using momentum (i.e. $m_p > 0$) performs better than not and it shows the inferior performance when the update is too slow ($m_p = 0.999$).

Loss	p_t	*Linear*	k-*NN*
CE	SM	61.9	58.5
	SK	63.1	60.1
ProtoCPC	SM	63.7	**60.9**
	SK	**63.9**	60.3

Table 3: Ablation on loss functions and p_t assignment methods. 'CE' for cross-entropy, 'SM' for softmax operator.

Teacher	Method	*Linear*	k-*NN*
SwAV	New P	60.8	54.5
	Old P	**63.1**	**57.7**
DINO	New P	**63.9**	**60.3**
	Old P	60.3	56.6

Table 4: Ablation on using teacher's old prototypes. 'Old P' for using teacher's old prototypes and 'New P' for copying prototypes.

m_p	*Linear*	k-*NN*
0.0	63.1	60.0
0.9	**63.9**	**60.3**
0.99	63.8	60.2
0.999	61.5	58.1

Table 5: Ablation on prior momentum rate m_p.

4 Discussion

Our ProtoCPC objective is shown to be effective in self-supervised representation distillation, suggesting possible application to other distillation tasks such as supervised model compression originated by knowledge distillation [9]. Note that previous contrastive objectives such as InfoNCE are concerned with critics modeled by inner product of two unit feature vectors, but one can use critics defined on the probability simplex as ProtoCPC does. The prototypical layer is imperative to match the probability space when using cross-entropy for measuring the discrepancy. One possible generalization can be done by using other probabilistic measures, such as Wasserstein distance, which is a metric and does not require two distributions to be on same probability space. We leave it for the future work.

References

[1] Zhiyuan Fang, Jianfeng Wang, Lijuan Wang, Lei Zhang, Yezhou Yang, and Zicheng Liu. Seed: Self-supervised distillation for visual representation. *arXiv preprint arXiv:2101.04731*, 2021.

[2] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[3] 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.

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

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

[6] Yuki Markus Asano, Christian Rupprecht, and Andrea Vedaldi. Self-labelling via simultaneous clustering and representation learning. *arXiv preprint arXiv:1911.05371*, 2019.

[7] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *arXiv preprint arXiv:2006.09882*, 2020.

[8] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *arXiv preprint arXiv:2104.14294*, 2021.

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

[10] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26:2292–2300, 2013.

[11] Jason Altschuler, Jonathan Weed, and Philippe Rigollet. Near-linear time approximation algorithms for optimal transport via sinkhorn iteration. *arXiv preprint arXiv:1705.09634*, 2017.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[15] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3733–3742, 2018.

[16] Michael Tschannen, Josip Djolonga, Paul K Rubenstein, Sylvain Gelly, and Mario Lucic. On mutual information maximization for representation learning. *arXiv preprint arXiv:1907.13625*, 2019.

[17] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pages 9929–9939. PMLR, 2020.

[18] Ibrahim Ahmad and Pi-Erh Lin. A nonparametric estimation of the entropy for absolutely continuous distributions (corresp.). *IEEE Transactions on Information Theory*, 22(3):372–375, 1976.

A Details on experiments

A.1 Teacher SSL models

The teacher SSL models are following:

- MoCo-v2 [4] uses InfoNCE loss and momentum encoder. Since InfoNCE requires large negatives, they pertain a large queue which updates by first-in-first-out rule with features of momentum encoder. We used the MoCo-v2 ResNet-50 [12] trained for 800 epochs.
- SwAV [7] is a prototypical method that generates probability by adopting prototypes. They generate probability of a given feature by computing similarity with respect to prototypes and minimize the cross-entropy between probability outputs of two different views of an image. They used Sinkhorn-Knopp iteration for probability output. Also, they first proposed multi-crops strategy, which additionally use small crops of an image to expedite the training. We used the SwAV ResNet-50 trained for 800 epochs with multi-crops applied.
- DINO [8] is a prototypical method which uses momentum encoder. The training progress is similar to SwAV except that they use momentum encoder on the prototypes and use online centering before the softmax operator. They showed that their method is effective in training vision transformer as well as convnet such as ResNet-50. We used the DINO ResNet-50 and DeiT-S/16 [14] trained for 800 epochs with multi-crops applied.

We archive the checkpoints of teacher models from the author's original implementation. We also conducted SSL on ResNet-18 with DINO which is the state-of-the-art method. We train ResNet-18 with same setting as the author did for ResNet-50, except that we used batch size of 512 and trained for 100 epochs.

A.2 Data augmentation

For data augmentation, the student networks are trained with same data augmentation as their teacher did. When using multi-crops data augmentation, we follow SwAV [7] for implementation. Note that in SEED [1], they conduct ablation studies on using the variety of views that student and teacher takes. They observed that using identical view for both student and teacher networks performed better than passing different views of an image to teacher and student. We follow this strategy, but for multi-crops data augmentation, we only compute the probability of teacher by one global view and this probability is passed to each probability of small local views. This counters the method in SEED.

A.3 Network architectures

We set the projection heads of student network to be same as the teacher network. When teacher is DINO DeiT, the teacher network do not contain batch normalization, but we add batch normalization to projection heads when training ResNet-18 student. Remark that the projection heads of MoCo and SwAV have output dimension of 128, and the projection head of DINO has output of 256. Then we set the number of prototypes to be $K = 65536$ throughout the experiments. For SwAV, since we use pre-trained prototypes, the number of prototypes is 3000. Every features are normalized before computation with prototypes, and prototypes are normalized during the training.

A.4 Training hyperparameters

For probability of teacher, we use SK operator with 3 steps of iteration and $\tau_t = 0.04$. For probability of student, we set $\tau_s = 0.1$. The prior momentum for ProtoCPC loss is 0.9. We use SGD optimizer with batch size 512 and weight decay is 1e-4. The learning rate is 0.6 and is decayed by cosine learning rate schedule to 1e-6.

A.5 Evaluation

For evaluation, we use both linear evaluation protocol and k-nearest neighbor classification. For linear evaluation protocol, we freeze the trained weight and train a linear classifier at the top of the frozen feature. We train with SGD optimizer with batch size 256 and use learning rate of 0.3 with

100 epochs. We use cosine learning rate decay schedule and we don't use weight decay. For k-NN classification, we follow weighted k-NN with $\tau = 0.07$ as done in [15].

B Discussion on ProtoCPC

B.1 Pseudocode for ProtoCPC loss

The PyTorch style pseudo-code for our ProtoCPC is demonstrated in Algorithm 1.

Algorithm 1 ProtoCPC loss PyTorch-style pseudocode.

```
# tps, tpt: student and teacher temperatures
# m: prior momentum rate
prior = torch.ones(1, K) # initialize prior with uniform
def ProtoCPC(zt, zs):
    zt = zt.detach()
    pt = SK(zt / tpt)
    zs = zs / tps

    prior = m*prior + (1-m)* K * torch.mean(pt, dim=0) # sum of prior is always K

    loss_align = -torch.sum(pt * zs, dim=1)
    loss_unif = torch.logsumexp(zs + torch.log(prior), 1)
    loss = loss_align + loss_unif
    return loss.mean()
```

B.2 Relationship with InfoNCE

While many lower bounds to the mutual information were proposed, [16] observe that the tightness of bound does not necessarily imply better representation learning performance. From then, many works focused on the analyzing the components of contrastive objective itself which are responsible for the empirical success. [17] argued that the contrastive loss is composed of alignment and uniformity loss, where alignment loss accounts for the similarity of two positive features and uniformity loss measures how the features are scattered in the unit hypersphere, and show that both losses are important in contrastive learning.

We draw analogy on ProtoCPC by dissecting into alignment and uniformity losses. Since the alignment loss is straightforward, we analyze the uniformity loss. Remark that one can interpret the uniformity loss $\mathcal{L}_{\text{ProtoCPC-Unif}}$ by the re-substitution entropy estimator of z_s via a von-Mises Fisher kernel density estimation (vMF-KDE) [18]:

$$\mathcal{L}_{\text{ProtoCPC-Unif}} = \mathbb{E}_{z_s \sim S}\left[\log \sum_{k=1}^{K} q^{(k)} \exp(\tilde{z}_s^{(k)}/\tau_s)\right] = \mathbb{E}_{z_s \sim S}\left[\log \sum_{k=1}^{K} q^{(k)} \exp(w_k \cdot z_s/\tau_s)\right] \quad (8)$$

$$= \mathbb{E}_{z_s \sim S}[\log \hat{p}_{\text{vMF-KDE}}(z_s)] + \log Z_{\text{vMF}} = -\hat{H}(z_s) + \log Z_{\text{vMF}}, \quad (9)$$

where each w_k is a k-th column of W_S and acts as a mean direction of k-th vMF distribution and $q^{(k)}$ acts as a prior for each k-th vMF distribution (It supports why q_k is called a prior). The $\hat{p}_{\text{vMF-KDE}}$ is thus the mixture of K vMF distribution with prior $q^{(k)}$ and then the uniformity loss is a re-substitution entropy $\hat{H}(z_s)$. The Z_{vMF} is a normalizing constant for vMF distribution. Remark that the uniformity loss of CPC objective is also a re-substitution entropy with vMF-KDE, but the mean directions are given by negative samples z_{tj} and the prior is uniform. It shows that the ProtoCPC objective allows modeling of complex mixture of vMF distribution by exploiting prior term and using prototypes remove the dependency on negative samples.