Clustering by Maximizing Mutual Information Across Views - Supplementary Material

Kien Do, Truyen Tran, Svetha Venkatesh
Applied Artificial Intelligence Institute (A2I2), Deakin University, Geelong, Australia
{k.do, truyen.tran, svetha.venkatesh}@deakin.edu.au

A. Appendix

A.1. Possible critics for the probability contrastive loss

We list here several possible critics that could be used in $\mathcal{L}_{\text{PC}}$. If we simply consider a critic f as a similarity measure of two probabilities p and q, f could be the *negative Jensen Shannon (JS) divergence*[1] between p and q:

$$f(p,q) = -D_{\text{JS}}(p\|q) \tag{1}$$

$$= -\frac{1}{2}\left(D_{\text{KL}}\left(p\|\frac{p+q}{2}\right) + D_{\text{KL}}\left(q\|\frac{p+q}{2}\right)\right) \tag{2}$$

or the *negative L2 distance* between p and q:

$$f(p,q) = -\left\|p-q\right\|_2^2 = -\sum_{c=1}^{C}\left(p_c - q_c\right)^2 \tag{3}$$

In both cases, f achieves its maximum value when $p = q$ and its minimum value when p and q are different one-hot vectors.

We can also define f as the *dot product* of p and q as follows:

$$f(p,q) = p^\top q = \sum_{c=1}^{C} p_c q_c \tag{4}$$

However, the maximum value of this critic is no longer obtained when $p = q$ but when p and q are *the same one-hot vector* (check Appdx. A.2 for details). It means that maximizing this critic encourages not only the consistency between p and q but also the confidence of p and q.

A.2. Global maxima and minima of the dot product critic for probabilities

Proposition 1. The dot product critic $f(p,q) = \sum_{c=1}^{C} p_c q_c$ achieves its global maximum value at 1 when p_c and q_c are *the same one-hot vector*, and its global minimum value at 0 when p_c and q_c are *different one-hot vector*.

[1]The JS divergence is chosen due to its symmetry. The negative sign reflects the fact that f is a similarity measure instead of a divergence.

Proof. Since $0 \leq p_c, q_c \leq 1$, we have $\sum_{c=1}^{C} p_c q_c \geq 0$. This minimum value is achieved when $p_c q_c = 0$ for all $c \in \{1, ..., C\}$. And because $\sum_{c=1}^{C} p_c = \sum_{c=1}^{C} q_c = 1$, p_c and q_c must be different one-hot vectors.

In addition, we also have $\sum_{c=1}^{C} p_c q_c \leq \sum_{c=1}^{C} p_c = 1$. This maximum value is achieved when $p_c q_c = p_c$ or $p_c(q_c - 1) = 0$ for all $c \in \{1, ..., C\}$, which means p_c and q_c must be the same one-hot vectors. □

Since the gradient of $\sum_{c=1}^{C} p_c q_c$ w.r.t. q_c is proportional to p_c, if we fix p and only optimize q, maximizing $\sum_{c=1}^{C} p_c q_c$ via gradient ascent will encourage q to be *one-hot* at the component k at which p_k is the *largest*. Similarly, minimizing $\sum_{c=1}^{C} p_c q_c$ via gradient descent will encourage q to be *one-hot* at the component k at which p_k is the *smallest*.

In case $p_1 = ... = p_C = \frac{1}{C}$, all the components of q have similar gradients. Although it does not change the relative order between the components of q after update, it still push q towards the saddle point $\left(\frac{1}{C}, ..., \frac{1}{C}\right)$. However, chance that models get stuck at this saddle point is tiny unless we explicitly force it to happen (e.g., maximizing $H(q)$).

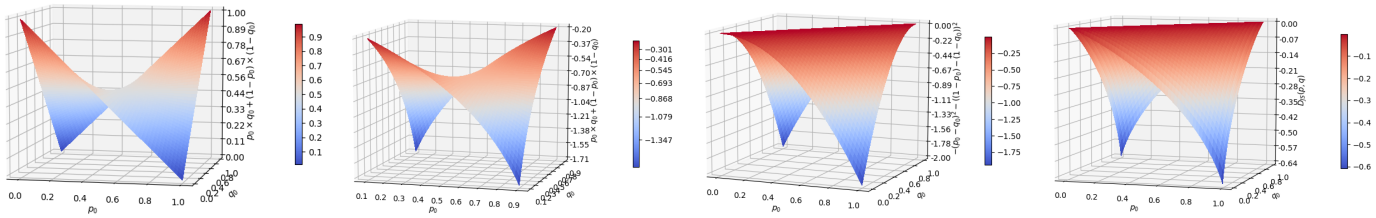
For better understanding of the optimization dynamics, we visualize the surface of $\sum_{c=1}^{C} p_c q_c$ with $C = 2$ in Fig. 1a. $\log\left(\sum_{c=1}^{C} p_c q_c\right)$ has the same global optimal values and surface as $\sum_{c=1}^{C} p_c q_c$

A.3. Derivation of the InfoNCE lower bound

The variational lower bound of $I(X;Y)$ can be computed as follows:

$$\begin{aligned} I(X;Y) &= \mathbb{E}_{p(x,y)}\left[\log\frac{p(x,y)}{p(x)p(y)}\right] \\ &= \mathbb{E}_{p(x,y)}\left[\log\frac{q_\theta(x,y)}{p(x)p(y)}\right] + D_{\text{KL}}\left(p(x,y)\|q_\theta(x,y)\right) \\ &\geq \mathbb{E}_{p(x,y)}\left[\log\frac{q_\theta(x,y)}{p(x)p(y)}\right] \end{aligned} \tag{5}$$

where $q_\theta(x,y)$ is the variational approximation of $p(x,y)$.



(a) Dot-product critic (b) Log-of-dot-product critic (c) Negative-L2-distance critic (d) Negative-JS-divergence critic

Figure 1: The surfaces of different critics on probabilities in case of 2 classes.

Following [10], we assume that $q_\theta(x, y)$ belongs to the energy-based variational family that uses a critic $f_\theta(x, y)$ and is scaled by the data density $p(x)p(y)$:

$$q_\theta(x, y) = \frac{p(x)p(y)e^{f_\theta(x,y)}}{\sum_{x,y} p(x)p(y)e^{f_\theta(x,y)}} = \frac{p(x)p(y)e^{f_\theta(x,y)}}{Z_\theta} \tag{6}$$

where $Z_\theta = \sum_{x,y} p(x)p(y)e^{f_\theta(x,y)} = \mathbb{E}_{p(x)p(y)}\left[e^{f_\theta(x,y)}\right]$ is the partition function which does not depend on x, y.

Since the optimal value of $q_\theta(x, y)$ is $q^*_\theta(x, y) = p(x, y)$, we have:

$$\begin{aligned} &\frac{p(x)p(y)e^{f^*_\theta(x,y)}}{Z^*_\theta} = p(x, y) \\ \Leftrightarrow &f^*_\theta(x, y) = \log Z^*_\theta + \log \frac{p(x, y)}{p(x)p(y)}, \end{aligned} \tag{7}$$

which means the optimal value of $f_\theta(x, y)$ is proportional to $\log \frac{p(x,y)}{p(x)p(y)}$.

Next, we will show that f_θ is the critic in the InfoNCE lower bound. We start by rewriting the lower bound in Eq. 5 using the formula of $q_\theta(x)$ in Eq. 6 as follows:

$$\begin{aligned} I(X; Y) &\geq \mathbb{E}_{p(x,y)}\left[\log \frac{e^{f_\theta(x,y)}}{Z_\theta}\right] \\ &= \mathbb{E}_{p(x,y)}\left[f_\theta(x, y)\right] - \log Z_\theta \end{aligned} \tag{8}$$

Here, we encounter the intractable $\log Z_\theta$. To form a tractable lower bound of $I(X; Y)$, we continue replacing $\log Z_\theta$ with its variational upper bound:

$$\log Z_\theta \leq \frac{Z_\theta}{a_\theta} + \log a_\theta - 1 \tag{9}$$

where a_θ is the variational approximation of Z_θ. We should choose a_θ close to Z_θ so that the variance of the bound in Eq. 9 is small. Recalling that $Z_\theta = \mathbb{E}_{p(x)p(y)}\left[e^{f_\theta(x,y)}\right]$, we define a_θ as follows:

$$a_\theta = \frac{1}{M}\sum_{i=1}^{M} e^{f_\theta(x_i,y)} \tag{10}$$

where $x_1, ..., x_M$ are M samples from $p(x)$. a_θ in Eq. 10 can be seen as a stochastic estimation of Z_θ with x sampled M times more than y. Thus, $\frac{Z_\theta}{a_\theta} \approx 1$ and from Eq. 9, we have $\log Z_\theta \leq \log a_\theta$. Apply this result to Eq. 8, we have:

$$I(X; Y) \geq \mathbb{E}_{p(x,y)}\left[f_\theta(x, y)\right] - \log a_\theta \tag{11}$$

$$= \mathbb{E}_{p(x_{2:M})}\mathbb{E}_{p(x_1,y)}\left[f_\theta(x_1, y) - \log \frac{1}{M}\sum_{i=1}^{M} e^{f_\theta(x_i,y)}\right] \tag{12}$$

$$= \mathbb{E}_{p(x_{1:M})p(y|x_1)}\left[\log \frac{e^{f_\theta(x_1,y)}}{\sum_{i=1}^{M} e^{f_\theta(x_i,y)}}\right] + \log M \tag{13}$$

$$\triangleq I_{\text{InfoNCE}}(X; Y) \tag{14}$$

where Eq. 12 is obtained from Eq. 11 by using the fact that $\mathbb{E}_{p(x,y)}\left[f_\theta(x, y)\right] = \mathbb{E}_{p(x_{2:M})}\mathbb{E}_{p(x_1,y)}\left[f_\theta(x_1, y)\right]$ and the assumption that the samples $x_1, ..., x_M$ and y in a_θ (Eq. 10) are drawn from $p(x_{2:M})p(x_1, y)$.

Combining with the result in Eq. 7, we have the optimal critic $f^*_\theta(x, y)$ in the InfoNCE lower bound is proportional to $\log \frac{p(x,y)}{p(x)p(y)} = \log \frac{p(y|x)}{p(y)}$. Since $p(y)$ does not depend on x and will be cancelled by both the nominator and denominator in Eq. 13, $f^*_\theta(x, y)$ is, in fact, proportional to $\log p(y|x)$.

A.4. Derivation of the scaled dot product critic in representation learning

Recalling that in contrastive representation learning, the critic f is defined as the scaled dot product between two unit-normed feature vectors $\tilde{z}$, z_i:

$$f(\tilde{x}, x_i) = \tilde{z}^\top z_i / \tau$$

Interestingly, this formula of f is accordant with the formula of f^* and is proportional to $\log p(\tilde{x}|x_i)$. To see why, let's assume that the distribution of $\tilde{z}$ given z_i is modeled by an isotropic Gaussian distribution with z_i as the mean

vector and τI as the covariance matrix. Then, we have:

$$\begin{aligned} f^* &\propto \log p(\tilde{x}|x_i) \\ &\approx \log p(\tilde{z}|z_i) \\ &\propto \log e^{-\frac{0.5}{\tau}\|\tilde{z}-z_i\|_2^2} \\ &= -\frac{0.5}{\tau}\left(\|\tilde{z}\|_2^2 - 2\tilde{z}^\top z_i + \|z_i\|_2^2\right) \\ &= \tilde{z}^\top z_i/\tau - 1/\tau \\ &\propto \tilde{z}^\top z_i/\tau \end{aligned}$$

where $\|\tilde{z}\|_2^2 = \|z_i\|_2^2 = 1$ due to the fact that $\tilde{z}$ and z_i are unit-normed vectors.

A.5. Analysis of the gradient of $\mathcal{L}_{\text{PC}}$

Recalling that the probability contrastive loss $\mathcal{L}_{\text{PC}}$ for a sample $\tilde{x}$ with the "log-of-dot-product" critic $f(p,q) = \log\left(p^\top q\right)$ is computed as follows:

$$\begin{aligned} \mathcal{L}_{\text{PC}} &= -\log \frac{e^{f(\tilde{q},q_1)}}{\sum_{i=1}^M e^{f(\tilde{q},q_i)}} \\ &= -\log\left(\tilde{q}^\top q_1\right) + \log \sum_{i=1}^M \tilde{q}^\top q_i \end{aligned}$$

Because $\tilde{q}$ is always parametric while q_i ($i \in \{1,...,M\}$) can be either parametric (if $\mathcal{L}_{\text{PC}}$ is implemented via the SimCLR framework [3]) or non-parametric (if $\mathcal{L}_{\text{PC}}$ is implemented via the MemoryBank framework [15]), we focus on the gradient of $\mathcal{L}_{\text{PC}}$ back-propagating through $\tilde{q}$. In practice, $\tilde{q}$ is usually implemented by applying softmax to the logit vector $\tilde{u} \in \mathbb{R}^C$:

$$\tilde{q}_c = \frac{\exp\left(\tilde{u}_c\right)}{\sum_{k=1}^C \exp\left(\tilde{u}_k\right)}$$

where $\tilde{q}_c$ denotes the c-th component of $\tilde{q}$. Similarly, $q_{i,c}$ is the c-th component of q_i.

The gradient of $\mathcal{L}_{\text{PC}}$ w.r.t. $\tilde{u}_c$ is given by:

$$\frac{\partial \mathcal{L}_{\text{PC}}}{\partial \tilde{u}_c} = -\frac{\partial}{\partial \tilde{u}_c}\log\left(\tilde{q}^\top q_1\right) + \frac{\partial}{\partial \tilde{u}_c}\log\sum_{i=1}^M \tilde{q}^\top q_i \qquad (15)$$

The first term in Eq. 15 is equivalent to:

$$\begin{aligned} & -\frac{\partial}{\partial \tilde{u}_c}\log\left(\tilde{q}^\top q_1\right) \\ \Leftrightarrow & \frac{-1}{\tilde{q}^\top q_1}\left(\frac{\partial}{\partial \tilde{u}_c}\left(\tilde{q}_c q_{1,c}\right) + \sum_{k\neq c}\frac{\partial}{\partial \tilde{u}_c}\left(\tilde{q}_k q_{1,k}\right)\right) \\ \Leftrightarrow & \frac{-1}{\tilde{q}^\top q_1}\left(\tilde{q}_c(1-\tilde{q}_c)q_{1,c} - \sum_{k\neq c}\tilde{q}_c\tilde{q}_k q_{1,k}\right) \\ \Leftrightarrow & -\frac{1}{\sum_{k=1}^C \tilde{q}_k q_{1,k}}\left(\tilde{q}_c q_{1,c} - \tilde{q}_c\sum_{k=1}^C \tilde{q}_k q_{1,k}\right) \\ \Leftrightarrow & \tilde{q}_c - \frac{\tilde{q}_c q_{1,c}}{\sum_{k=1}^C \tilde{q}_k q_{1,k}} \end{aligned} \qquad (16)$$

And the second term in Eq. 15 is equivalent to:

$$\begin{aligned} & \frac{\partial}{\partial \tilde{u}_c}\log\sum_{i=1}^M \tilde{q}^\top q_i \\ \Leftrightarrow & \frac{1}{\sum_{i=1}^M \tilde{q}^\top q_i}\left(\sum_{i=1}^M \frac{\partial}{\partial \tilde{u}_c}\left(\tilde{q}^\top q_i\right)\right) \\ \Leftrightarrow & \frac{1}{\sum_{i=1}^M \tilde{q}^\top q_i}\left(\sum_{i=1}^M\left(\tilde{q}_c q_{i,c} - \tilde{q}_c\sum_{k=1}^C \tilde{q}_k q_{i,k}\right)\right) \\ \Leftrightarrow & \frac{1}{\sum_{i=1}^M\sum_{k=1}^C \tilde{q}_k q_{i,k}}\left(\sum_{i=1}^M \tilde{q}_c q_{i,c} - \tilde{q}_c\sum_{i=1}^M\sum_{k=1}^C \tilde{q}_k q_{i,k}\right) \\ \Leftrightarrow & \frac{\sum_{i=1}^M \tilde{q}_c q_{i,c}}{\sum_{i=1}^M\sum_{k=1}^C \tilde{q}_k q_{i,k}} - \tilde{q}_c \end{aligned}$$

Thus, we have:

$$\frac{\partial \mathcal{L}_{\text{PC}}}{\partial \tilde{u}_c} = \frac{\sum_{i=1}^M \tilde{q}_c q_{i,c}}{\sum_{i=1}^M\sum_{k=1}^C \tilde{q}_k q_{i,k}} - \frac{\tilde{q}_c q_{1,c}}{\sum_{k=1}^C \tilde{q}_k q_{1,k}} \qquad (17)$$

We care about the second term in Eq. 17 which is derived from the gradient of the critic $f(\tilde{q}, q_1)$ w.r.t. $\tilde{u}_c$ (the negative of the term in Eq. 16). We rewrite this gradient with simplified notations as follows:

$$\frac{\partial f(q,p)}{\partial u_c} = \frac{q_c p_c}{\sum_{k=1}^C q_k p_k} - q_c$$

where u_c is the c-th logit of q. Since during training, q is encouraged to be one-hot (see Appdx. A.2), the denominator may not be defined if we do not prevent p from being a different one-hot vector. However, even when the denominator is defined, the update still does not happen as expected when q is one-hot. To see why, let's consider a simple scenario in which $q = [0,1,0]$ and $p = [0.998, 0.001, 0.001]$. Apparently, the denominator is $0.001 \neq 0$. By maximizing

Dataset	#Train	#Test	#Extra	#Classes	Image size
CIFAR10	50,000	10,000	×	10	32×32×3
CIFAR20	50,000	10,000	×	20	32×32×3
STL10	5,000	8,000	100,000	10	96×96×3
ImageNet10	13,000	500	×	10	224×224×3
ImageNet-Dogs	19,500	750	×	15	224×224×3
ImageNet-50	64,274	2,500	×	50	224×224×3
ImageNet-100	128,545	5,000	×	100	224×224×3
ImageNet-200	256,558	10,000	×	200	224×224×3

Table 1: Details of the datasets used in this work.

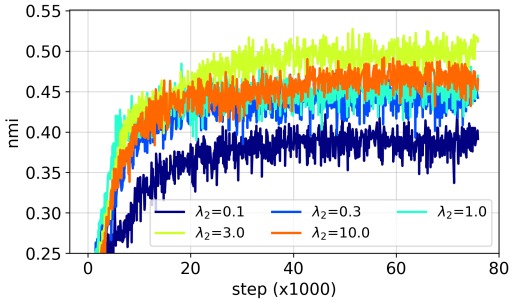


Figure 2: NMI curve of CRLC on ImageNet-Dogs w.r.t. different coefficients of $\mathcal{L}_{\text{FC}}$.

$f(q,p)$, we want to push q toward p. Thus, we expect that $\frac{\partial f}{\partial u_1} > 0$ and $\frac{\partial f}{\partial u_2} < 0$. However, the gradients w.r.t. u_c are 0s for all $c \in \{1,2,3\}$:

$$\frac{\partial f}{\partial u_1} = \frac{0 \times 0.998}{0.001} - 0 = 0$$
$$\frac{\partial f}{\partial u_2} = \frac{1 \times 0.001}{0.001} - 1 = 0$$
$$\frac{\partial f}{\partial u_3} = \frac{0 \times 0.001}{0.001} - 0 = 0$$

The reason is that $q = [0,1,0]$ is a stationary point (minimum in this case). This means once the model has set q to be one-hot, it tends to get stuck there and cannot escape *regardless of the value of* p. This problem is known in literature as the "saturating gradient" problem. To alleviate this problem, we propose to smooth out the values of q and p before computing the critic f:

$$q = (1-\gamma)q + \gamma r$$
$$p = (1-\gamma)p + \gamma r$$

where $0 \leq \gamma \leq 1$ is the smoothing coefficient, which is set to 0.01 if not otherwise specified; $r = (\frac{1}{C}, ..., \frac{1}{C})$ is the uniform probability vector over classes. We also regularize the value of u_c to be within $[-25, 25]$.

A.6. Dataset description

In Table 1, we provide details of the datasets used in this work. CIFAR20 is CIFAR100 with 100 classes replaced by 20 super-classes. STL10 is different from other datasets in the sense that it has an auxiliary set of 100,000 unlabeled samples of unknown classes. Similar to previous works, we use samples from this auxiliary set and the training set to train the "representation learning" head.

A.7. Training setups for clustering

End-to-end clustering For end-to-end clustering, we use a SGD optimizer with a constant learning rate = 0.1, momentum = 0.9, Nesterov = False, and weight decay = 5e-4 based on the settings in [5, 6, 13]. We set the batch size to 512 and the number of epochs to 2000. In fact, on some datasets like ImageNet10 or ImageNet-Dogs, CRLC only needs 500 epochs to converge. The coefficients of the negative entropy and $\mathcal{L}_{\text{FC}}$ (λ_1 and λ_2 in Eq. 11 in the main text) are fixed at 1 and 10, respectively. Each experiment is repeated 3 times with random initializations.

Two-stage clustering For two-stage clustering, we use the same settings as in [14]. Specifically, the backbone network is ResNet18 for CIFAR10/20, STL10 and is ResNet50 for ImageNet50/100/200. In the first (pretraining) stage, for CIFAR10/20 and STL10, we pretrain the backbone network and the RL-head via SimCLR [3] for 500 epochs. The optimizer is SGD with an initial learning rate = 0.4 decayed with a cosine decay schedule [9], momentum = 0.9, Nesterov = False, and weight decay = 1e-4. Meanwhile, for ImageNet50/100/200, we directly copy the pretrained weights of MoCo [6] to the backbone network and the RL-head. After the pretraining stage, we find for each sample in the training set 50 nearest neighbors based on the cosine similarity measure. Positive samples for contrative learning in the second stage are drawn uniformly from these sets of nearest neighbors. In the second stage, for CIFAR10/20 and STL10, we train both the backbone network and the C-head for 200 epochs by minimizing $\mathcal{L}_{\text{cluster}}$ (Eq. 8 in the main text) using an Adam optimizer with a constant learning rate = 1e-4 and weight decay = 1e-4. For ImageNet50/100/200, we freeze the backbone network and only train the C-head for 200 epochs by minimizing $\mathcal{L}_{\text{cluster}}$ using an SGD optimizer with a constant learning rate = 5.0, momentum = 0.9, Nesterov = False, and weight decay = 0.0.

A.8. Complete end-to-end clustering results

Complete results with standard deviations on the five standard clustering datasets are shown in Tables 2 and 3. From Table 2, we see that for CIFAR10 using both the training and test sets does not cause much difference in performance compared to using only the training set. For CI-

Dataset		CIFAR10			CIFAR20			STL10		
Metric		ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI
C-head only	Train only	**67.2±0.7**	56.8±1.3	**47.8±1.4**	**38.0±1.6**	**36.8±1.0**	**22.3±0.9**	47.03±2.2	39.06±1.5	27.23±1.8
	Train + Test	66.9±0.8	**56.9±0.7**	47.5±0.5	37.7±0.4	35.7±0.5	21.6±0.3	**61.2±1.2**	**52.7±0.8**	**43.4±1.3**
CRLC	Train only	79.4±0.3	66.7±0.6	62.3±0.4	**43.4±0.8**	**43.1±0.5**	**27.7±0.3**	57.6±1.6	50.8±1.5	41.9±1.2
	Train + Test	**79.9±0.6**	**67.9±0.6**	**63.4±0.4**	42.5±0.7	41.6±0.8	26.3±0.5	**81.8±0.3**	**72.9±0.4**	**68.2±0.3**

Table 2: Clustering results of our proposed methods on CIFAR10, CIFAR20 and STL10 with only the training set used and with both the training and test sets used.

Dataset	ImageNet10			ImageNet-Dogs		
Metric	ACC	NMI	ARI	ACC	NMI	ARI
C-head only	80.0±1.4	75.2±1.9	67.6±2.2	36.3±0.9	37.5±0.7	19.8±0.4
CRLC	**85.4±0.3**	**83.1±0.5**	**75.9±0.4**	**46.1±0.6**	**48.4±0.6**	**29.7±0.4**

Table 3: Clustering results of our proposed methods on ImageNet10 and ImageNet-Dogs.

FAR20, using only the training set even leads to slightly better results. By contrast, for STL10, models trained with both the training and test sets significantly outperform those trained with the training set only. We believe the reason is that for CIFAR10 and CIFAR20, the training set is big enough to cover the data distribution in the test set while for STL10, it does not apply (Table 1). Therefore, we think subsequent works should use only the training set when doing experiments on CIFAR10 and CIFAR20.

A.9. Additional two-stage clustering results

Table 4 compares the clustering results of "two-stage" CRLC and SCAN on CIFAR10/20, STL10. "Two-stage" CRLC clearly outperforms SCAN on all datasets.

A.10. Additional ablation study results

A.10.1 Contribution of the feature contrastive loss

In Fig. 2, we show the performance of CRLC on ImageNet-Dogs w.r.t. different coefficients of $\mathcal{L}_{\text{FC}}$ (λ_2 in Eq. 11 in the main text). We observe that CRLC achieves the best clustering accuracy when $\lambda_2 = 3$. However, in Table 1 in the main text, we still report the result when $\lambda_2 = 10$.

A.10.2 Nonparametric implementation of CRLC

In this section, we empirically investigate the contributions of the number of negative samples and the momentum coefficient (α in Eq. 10 in the main text) to the performance of MemoryBank-based CRLC.

Contribution of the number of negative samples From Fig. 3a, we do not see any correlation between the number of negative samples and the clustering performance of MemoryBank-baed CRLC despite the fact that increasing the number of negative samples allows the RL-head and the C-head to gain more information from data (Figs. 3b and 3c). It suggests that for clustering (and possibly other classification tasks), getting more information may not lead to good results. Instead, we need to extract the right information related to clusters.

Contribution of the momentum coefficient From Fig. 4b, we see that changing the momentum value for updating probability vectors stored in the memory bank does not affects amount of information captured by the RL-head much. By contrast, in Fig. 4c, we see that larger values of the momentum cause the C-head to capture more information. This is reasonable because the accumulated probability vector $q_{n,t}$ is usually more stochastic (contains more information) than the probability vector $\hat{q}_n$ of a particular view (Eq. 10 in the main text). Larger values of the momentum also cause the model to converge slower but do not affect the performance much (Fig. 4a).

A.11. Qualitative evaluation

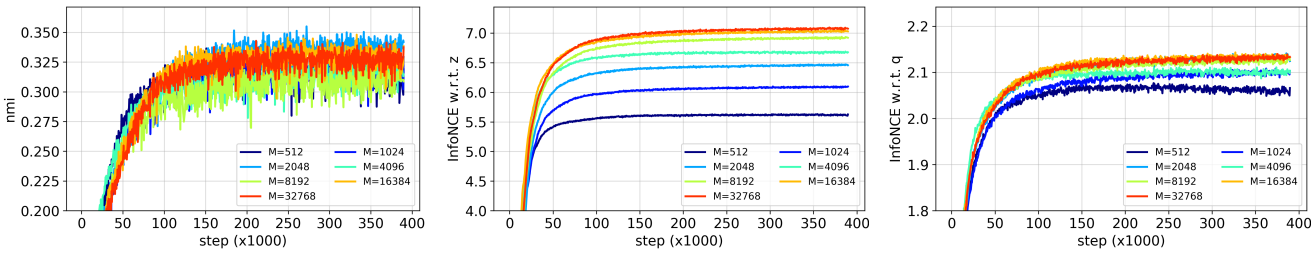
In Fig. 5, we show the top correctly predicted samples according to their confidence score for each of 5 classes from the training set of STL10. It is clear that these samples are representative of the cluster they belong to.

A.12. Consistency-regularization-based semi-supervised learning methods

When some labeled data are given, the clustering problem naturally becomes semi-supervised learning (SSL). The core idea behind recent state-of-the-art SSL methods such as UDA [16], MixMatch [2], ReMixMatch [1], FixMatch [11] is *consistency regularization* (CR) which is about forcing an input sample under different perturbations/augmentations to have similar class predictions. In

Dataset	CIFAR10			CIFAR20			STL10		
Metric	ACC	NMI	ARI	ACC	NMI	ARI	ACC	NMI	ARI
K-means [14]	65.9±5.7	59.8±2.0	50.9±3.7	39.5±1.9	40.2±1.1	23.9±1.1	65.8±5.1	60.4±2.5	50.6±4.1
SCAN [14]	81.8±0.3	71.2±0.4	66.5±0.4	42.2±3.0	44.1±1.0	26.7±1.3	75.5±2.0	65.4±1.2	59.0±1.6
two-stage CRLC	**84.2±0.1**	**74.7±0.3**	**70.6±0.5**	**45.0±0.7**	**44.8±0.8**	**28.7±0.9**	**78.7±1.1**	**68.4±1.6**	**62.7±1.8**

Table 4: Two-stage clustering results on CIFAR10/20 and STL10.



(a) NMI (b) InfoNCE w.r.t. $\mathcal{L}_{\text{FC}}$ (c) InfoNCE w.r.t. $\mathcal{L}_{\text{PC}}$

Figure 3: Learning curves of MemoryBank-based CRLC on CIFAR20 w.r.t. different numbers of negative samples. The momentum is $\alpha = 0.5$. The InfoNCE w.r.t. a contrastive loss is computed by using Eq. 2 in the main text.

this sense, CR can be seen as an unnormalized version of the probability contrastive loss without the denominator. Different SSL methods extend CR in different ways. For example, UDA uses strong data augmentation to generate positive pairs. MixMatch and ReMixMatch combines CR with MixUp [17]. However, none of the above methods achieve consistent performance with extremely few labeled data (Section 5.2 in the main text). By contrast, clustering methods like CRLC perform consistently well even when no label is available. Thus, we believe designing a method that enjoys the strength of both fields is possible and CRLC-semi can be one step towards that goal.

A.13. Training setups for semi-supervised learning

To train CRLC-semi, we use a SGD optimizer with an initial learning rate = 0.1, momentum = 0.9, Nesterov = False, and weight decay = 5e-4. Similar to [11], we adjust the learning rate at each epoch using a cosine decay schedule [9] computed as follows:

$$\text{lr}_t = \text{lr}_{\text{min}} + (\text{lr}_{\text{init}} - \text{lr}_{\text{min}}) \times \frac{1 + \cos\left(\frac{t}{T}\pi\right)}{2}$$

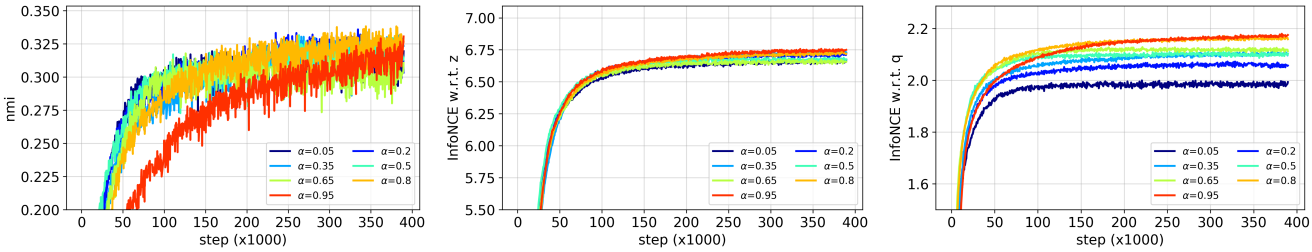
where $\text{lr}_{\text{init}} = 0.1$, $\text{lr}_{\text{min}} = 0.001$, lr_t is the learning rate at epoch t over T epochs in total. T is 2000 and 1000 for CIFAR10 and CIFAR100, respectively. The number of labeled and unlabeled samples in each batch is 64 and 512, respectively. In $\mathcal{L}_{\text{CRLC-semi}}$ (Eq. 12 in the main text), $\lambda_1 = 1$, $\lambda_2 = 5$, and $\lambda_3 = 1$.

We reimplement FixMatch using sample code from Github[2] with the default settings unchanged. In this code, the number of labeled and unlabeled data in a batch is 64 and 448, respectively. However, the number of steps in one epoch does not depend on the batch size but is fixed at 1024. Thus, FixMatch is trained in 1024 epochs $\approx$ 1 million steps for both CIFAR10 and CIFAR100. Meanwhile, CLRC-semi is trained in only 194,000 steps for CIFAR10 and 97,000 steps for CIFAR100.

A.14. More results on semi-supervised learning

In Table 5, we show additional semi-supervised learning results of CRLC-semi on CIFAR10 and CIFAR100 in comparison with more baselines. CRLC-semi clearly outperforms all standard baselines like Π-model, Pseudo Labeling or Mean Teacher. However, CRLC-semi looses its advantage over holistic methods like MixMatch [2] and methods that use strong data augmentation like UDA [16] or ReMixMatch [1] when the number of labeled data is big enough. Currently, we are not sure whether the problem comes from the feature contrastive loss $\mathcal{L}_{\text{FC}}$ (when we have enough labels, representation learning may act as a regularization term and reduce the classification result), or from the negative entropy term in $\mathcal{L}_{\text{cluster}}$ (causing too much regularization), or even from the probability contrastive loss (contrasting probabilities of two related views is not suitable when we have enough labels). Thus, we leave the answer of this question for future work. To gain more insight about the advantages of our proposed CRLC-semi, we provide detailed comparison between this method and the best

[2] https://github.com/CoinCheung/fixmatch-pytorch



(a) NMI (b) InfoNCE w.r.t. $\mathcal{L}_{\text{FC}}$ (c) InfoNCE w.r.t. $\mathcal{L}_{\text{PC}}$

Figure 4: Learning curves of MemoryBank-based CRLC on CIFAR20 w.r.t. different values of the momentum. The number of negative samples is $M = 4096$. The InfoNCE w.r.t. a contrastive loss is computed by using Eq. 2 in the main text.

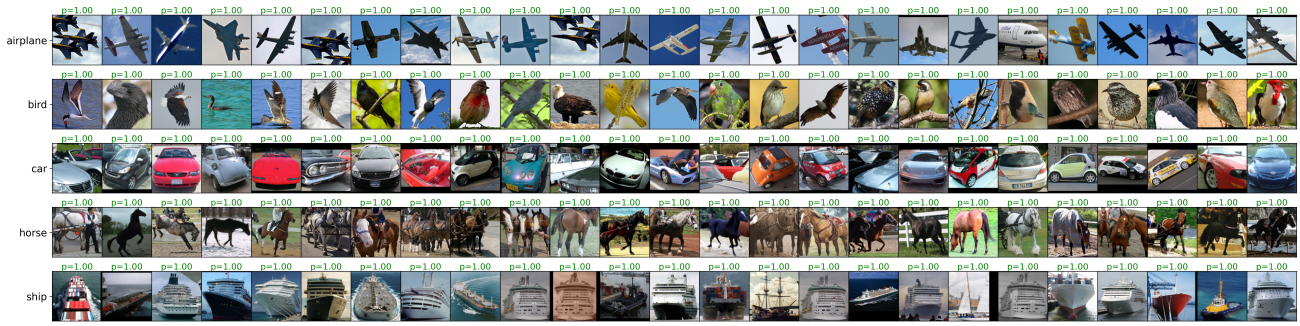


Figure 5: STL10 samples of 5 classes correctly predicted by CRLC. Samples are sorted by their confidence scores.

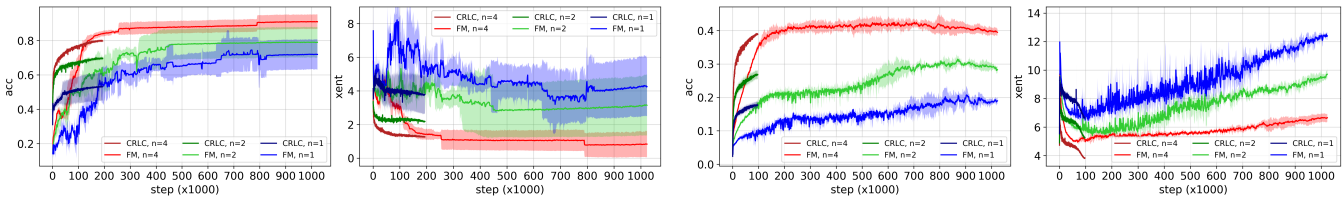
SSL baseline - FixMatch [11] in the next section.

Direct comparison between CRLC-semi and FixMatch FixMatch [11] is a powerful SSL method that makes use of pseudo-labeling [8] and strong data augmentation [4] to generate quality pseudo-labels for training. FixMatch has been shown to work reasonably well with only 1 labeled sample per class. In our experiment, we observe that FixMatch outperforms CRLC-semi on both CIFAR10 and CIFAR100. However, FixMatch must be trained in much more steps than CRLC-semi to achieve good results and its performance is very inconsistent (like other SSL baselines) compared to that of CRLC-semi (Figs. 6, 7).

Details of the labeled samples For the purpose of comparison and reproducing the results in Table 5, we provide the indices of 40 labeled CIFAR10 samples and 400 labeled CIFAR100 samples used in our experiments in Fig. 9 and Fig. 11, respectively. We also visualize these samples in Fig. 8 and 10. We note that we do not cherry-pick these samples but randomly draw them from the training set.

Dataset	CIFAR10				CIFAR100			
Labels	10	20	40	250	100	200	400	2500
Π-model [7]	-	-	-	54.26±3.97	-	-	-	57.25±0.48
Pseudo Labeling [8]	-	-	-	49.78±0.43	-	-	-	57.38±0.46
Mean Teacher [12]	-	-	-	32.32±2.30	-	-	-	53.91±0.57
MixMatch [2]	-	-	47.54±11.50	11.05±0.86	-	-	67.61±1.32	39.94±0.37
UDA [16]	-	-	29.05±5.93	8.82±1.08	-	-	59.28±0.88	33.13±0.22
ReMixMatch [1]	-	-	19.10±9.64	5.44±0.05	-	-	44.28±2.06	27.43±0.31
FixMatch (RA) [11]	-	-	13.81±3.37	5.07±0.65	-	-	**48.85±1.75**	28.29±0.11
ReMixMatch†[3]	59.86±9.34	41.68±8.15	28.31±6.72	-	**76.32±4.30**	**66.51±2.86**	52.23±1.71	-
FixMatch (RA)†[4]	**25.49±7.74**	**21.15±8.96**	**8.87±4.29**	-	79.27±2.65	68.58±0.7	57.52±1.5	-
CRLC-semi	46.75±8.01	29.81±1.18	19.87±0.82	13.53±0.21	82.20±1.15	73.04±1.15	60.87±0.17	41.10±0.12

Table 5: Full classification errors on CIFAR10 and CIFAR100. Lower values are better. Results of baselines are taken from [11]. †: Results obtained from external implementations of models.



(a) CIFAR10 (b) CIFAR100

Figure 6: Test accuracy and crossentropy curves of CRLC-semi (CRLC) and FixMatch (FM) on CIFAR10 and CIFAR100 with 1, 2, 4 labeled samples per class. It is clear that CRLC-semi performs consistently in all cases except for the case of CIFAR10 with 1 labeled sample per class. However, even in that case, the CRLC-semi still gives consistent performance for each run (Fig. 7). FixMatch, by contrast, is very inconsistent in its performance for each run, especially on CIFAR10.

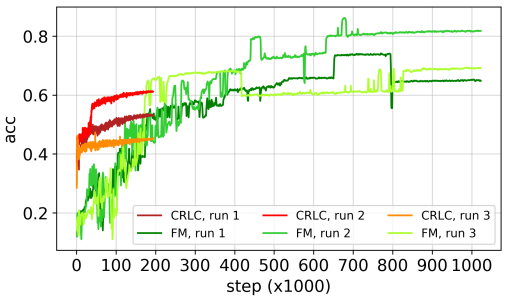


Figure 7: Test accuracy curves of CRLC-semi (CRLC) and FixMatch (FM) on CIFAR10 with 1 labeled samples per class w.r.t. 3 different runs.

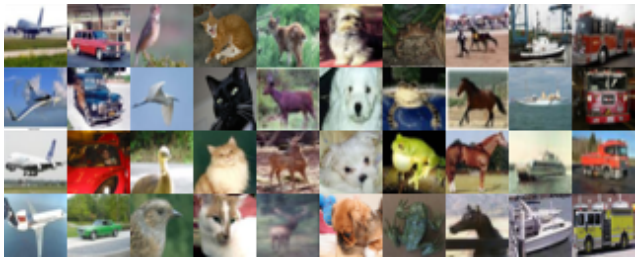


Figure 8: 40 labeled CIFAR10 samples organized into 4 rows where each row has 10 images corresponding to 10 classes. For 10 and 20 labeled samples, the first row and the first two rows are considered, respectively.

```
[[33797 42143 20308 23202 39495 37706 17788 22128 38925  5884]
 [ 2804 39911  6041 11188 20588 33193 16982 15878 42066 27972]
 [19066  2339 24978  1098 12132 15219 14139  2358 40495 37444]
 [19065 19165 16050 31194  3377 26529 22764  7989 14979 43282]]
```

Figure 9: Indices in the training set of the images in Fig. 8

Figure 10: 400 labeled CIFAR100 samples organized into 4 image blocks where each image block is a set of 100 images corresponding to 100 classes. For 100 and 200 labeled samples, the first block and the first two blocks are considered, respectively.

Block 1

```
[[11188 12218  6223 32575 15073 31887 46913 24978 26529 14442]
 [29329 38925 42143  9627 17117 26223 49586 15463 14283 21116]
 [14139  9544 25304 44940 23202 49718  3328  1538 42066 19066]
 [35592 39911 26534 47536  5884 28737 31867 10818  2363 24205]
 [48129 14360  2339 30952 19165 16982 39711 39354 41086 10609]
 [17925  6041 17788 40459 14979 11003 25059 49750 20308 38061]
 [ 1335 13367 27767 33797 19065 17978 46845  1088  3377 24528]
 [37444 13803 15977  4794 15219 39495 31626 48985 12344 20588]
 [45896 35097 31194 48299 27972 40517 23900 45209  3336 33193]
 [43282 37265 40495 46028 16050  8935 38158 43907  8983 24193]]
```

Block 2

```
[[39404 41352 37487 21791 24545 33045 39512 35960 33548 35465]
 [32244 22764 42462 11395 10836 14064 20797 15878 37129  8097]
 [42221 47066 46915  1788 11672 41659 35411 42141 46765  3788]
 [11894 21483 21743 16576 30846 39504 43770 26677 47042 49729]
 [23392  2732 16269 22389 47738 32627  4859 49852 20985 34982]
 [ 4885 12132 11095 29010  6592 28341 15536 38534 37706 10927]
 [ 3388 43257  8092 44052  7783  8225  9025 25138 25540 34907]
 [29427 36540 17999 18832  8423  1045 44302 37176  5845 24493]
 [25850 24481 43866 42061 42445 30962  2235 42427 17239  1791]
 [32694 48503  2804 26725 20441 42567 48444 37047 26901 13813]]
```

Block 3

```
[[36391 42934 35048 13579  8292 45480 42748 45984 11381 46018]
 [25501 35130 22658  5243  4287 32594 38519  7989 36761  4356]
 [25513 28523  7341 26116 10648 16563 20562  9467 42004 35726]
 [46746 21177 42454 15881 26838 24142 47376  8800 34485 33238]
 [12220  1216 34677 15429 35645 34202 43344 14026  1170 34224]
 [27277  1098    27 30163 48136 31554 36374 13139 21529  4708]
 [21360 11965  1695 37345  3968 36877  2358 34036 45044 31733]
 [45776 48496 34381 44941  8407 47812 11034 43694 47371 21209]
 [13998 43513 39228  3563 44723 16640 38753 16465 38529 40484]
 [26700 39133 36219 31865 49277 46387 19415 17068 26761 14239]]
```

Block 4

```
[[30442 39988 32270  6709  7017 26062 47575 20284 20982 31699]
 [29045 14242 36386 18365 35688 19535 39851  3473 15969 35477]
 [ 2653 44887 37250 45939 27313  5377  7564 35108 38461  2881]
 [24036 19749 16007 30737  2324 21277 38917 40713 25945 33506]
 [ 3671 41641 16667 30119 30028 19345 15737  5637 44468 24588]
 [32841 22128 37091 20991 10893 47385 38346  3399  3159 18757]
 [14970 46469  4780 26897 31836 31718 15863  2906 48034 19203]
 [ 5679 30322 39035 28835 21763 39729 28298 46213 32227  4517]
 [28894 22504 19817 27575 37802 31236  5186 29915 39333  5896]
 [37311 11193 32734 42035 10314  4613 34299 26479  3272 41338]]
```

Figure 11: Indices in the training set of the images in Fig. 10

References

[1] David Berthelot, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. *arXiv preprint arXiv:1911.09785*, 2019. 5, 6, 8

[2] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems*, pages 5050–5060, 2019. 5, 6, 8

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020. 3, 4

[4] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical data augmentation with no separate search. *arXiv preprint arXiv:1909.13719*, 2019. 7

[5] Prasoon Goyal, Zhiting Hu, Xiaodan Liang, Chenyu Wang, Eric P Xing, and Carnegie Mellon. Nonparametric variational auto-encoders for hierarchical representation learning. In *ICCV*, pages 5104–5112, 2017. 4

[6] 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

[7] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242*, 2016. 8

[8] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 2, 2013. 7, 8

[9] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 4, 6

[10] Ben Poole, Sherjil Ozair, Aaron Van Den Oord, Alex Alemi, and George Tucker. On variational bounds of mutual information. In *International Conference on Machine Learning*, pages 5171–5180, 2019. 2

[11] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *arXiv preprint arXiv:2001.07685*, 2020. 5, 6, 7, 8

[12] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems*, pages 1195–1204, 2017. 8

[13] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding. *arXiv preprint arXiv:1906.05849*, 2019. 4

[14] Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Scan: Learning to classify images without labels. In *European Conference on Computer Vision*, pages 268–285. Springer, 2020. 4, 6

[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. 3

[16] Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. Unsupervised data augmentation for consistency training. 2019. 5, 6, 8

[17] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. 6