Parametric Contrastive Learning
Supplementary Material

A. Proof to Remark 1

For an image X_i and its label y_i, the expectation number of positive pairs with respect to X_i will be:

$$K_{y_i} = q(y_i) * (\texttt{length}(queue) + batchsize * 2 - 1) \approx \texttt{length}(queue) \cdot q(y_i), \tag{9}$$

$q(y_i)$ is the class frequency over the whole dataset. Here the "$\approx$" establishes because $batchsize \ll \texttt{length}(queue)$ in training process. Note that we use such approximation just for simplification. Our analysis holds for the precise K_{y_i}. In what follows, we prove the optimal values for supervised contrastive loss.

Suppose training samples are i.i.d. To minimize the supervised contrastive loss for sample X_i, according to Eq. (3), we rewrite:

$$\begin{cases} P(i) = & \{z_1^+, z_2^+, ..., z_{K_{y_i}}^+\}; \\ p_i^+ = & \dfrac{\exp(z_i^+ \cdot T(x_i))}{\sum_{z_k \in A(i)} \exp(z_k \cdot T(x_i))}; \\ p_{sum}^+ = & p_1^+ + p_2^+ + ... + p_{K_{y_i}}^+. \end{cases}$$

Then the supervised contrastive loss will be:

$$\begin{aligned} \mathcal{L}_i &= -\sum_{z_+ \in P(i)} \log \frac{\exp(z_+ \cdot T(x_i))}{\sum_{z_k \in A(i)} \exp(z_k \cdot T(x_i))} \\ &= -(\log p_1^+ + \log p_2^+ + ... + \log p_{K_{y_i}}^+). \end{aligned}$$

For obtaining its optimal solution, we define the Lagrange multiplier form of $\mathcal{L}_i$ as:

$$l = -(\log p_1^+ + \log p_2^+ + ... + \log p_{K_{y_i}}^+) + \lambda(p_1^+ + p_2^+ + ... + p_{K_{y_i}}^+ - p_{sum}^+), \tag{10}$$

where λ is the Lagrange multiplier. The first order conditions of Eq. (10) w.r.t. λ and p_i^+ can be written as follows:

$$\begin{cases} \dfrac{\partial l}{\partial p_i^+} = -\dfrac{1}{p_i^+} + \lambda = 0; \\ \dfrac{\partial l}{\partial \lambda} = p_1^+ + p_2^+ + ... + p_{K_{y_i}}^+ - p_{sum}^+ = 0. \end{cases} \tag{11}$$

From Eq. (11), the optimal solution for p_i^* will be $\dfrac{p_{sum}^+}{K_{y_i}}$. Note that $p_{sum}^+ \in [0, 1]$, with a specific p_{sum}^+, the minimal loss value of $\mathcal{L}_i$ is:

$$\mathcal{L}_i = -K_{y_i} \log \frac{p_{sum}^+}{K_{y_i}}. \tag{12}$$

Thus, when $p_{sum}^+ = 1.0$, $\mathcal{L}_i$ achieves minimum with the optimal value $p_i^+ = \dfrac{1}{K_{y_i}}$ which is exactly the probability that two samples of the same class are a true positive pair.

B. Proof to Remark 2

For the image X_i and its label y_i, Eq. (9) still establishes for our parametric contrastive loss. To minimize the parametric contrastive loss for sample X_i, according to Eq. (4), we similarly rewrite:

$$\begin{cases} P(i) = & \{z_1^+, z_2^+, ..., z_{K_{y_i}}^+\} \\ p_i^+ = & \frac{\exp(z_i^+ \cdot T(x_i))}{\sum_{z_k \in A(i) \cup \mathbf{C}} \exp(z_k \cdot T(x_i))} \\ p_c^+ = & \frac{exp(c_y \cdot T(x_i))}{\sum_{z_k \in A(i) \cup \mathbf{C}} \exp(z_k \cdot T(x_i))} \\ p_{sum}^+ = & p_1^+ + p_2^+ + ... + p_{K_{y_i}}^+ + p_c^+. \end{cases}$$

Then the parametric contrastive loss will be:

$$\begin{aligned} \mathcal{L}_i &= \sum_{z_+ \in P(i) \cup \{c_y\}} -w(z_+) \log \frac{\exp(z_+ \cdot T(x_i))}{\sum_{z_k \in A(i) \cup \mathbf{C}} \exp(z_k \cdot T(x_i))} && (13) \\ &= -\left(\log p_c^+ + \alpha \cdot (\log p_1^+ + \log p_2^+ + ... + \log p_{k_{y_i}}^+)\right). && (14) \end{aligned}$$

For obtaining its optimal solution, we define the Lagrange multiplier form of $\mathcal{L}_i$ as:

$$l = -\left(\log p_c^+ + \alpha \cdot (\log p_1^+ + \log p_2^+ + ... + \log p_{k_{y_i}}^+)\right) + \lambda(p_1^+ + p_2^+ + ... + p_{K_{y_i}}^+ + p_c^+ - p_{sum}^+), \quad (15)$$

where λ is the Lagrange multiplier. The first order conditions of Eq. (15) w.r.t. λ, p_c^+ and p_i^+ can be written as follows:

$$\begin{cases} \frac{\partial l}{\partial p_i^+} = -\frac{\alpha}{p_i^+} + \lambda = 0; \\ \frac{\partial l}{\partial p_c^+} = -\frac{1}{p_c^+} + \lambda = 0; \\ \frac{\partial l}{\partial \lambda} = p_1^+ + p_2^+ + ... + p_{K_{y_i}}^+ + p_c^+ - p_{sum}^+ = 0. \end{cases} \quad (16)$$

From Eq. (16), the optimal solution for p_i^+ and p_c^+ will be $\frac{\alpha p_{sum}^+}{1+\alpha K_{y_i}}$ and $\frac{p_{sum}^+}{1+\alpha K_{y_i}}$ respectively. Note that $p_{sum}^+ \in [0, 1]$, with a specific p_{sum}^+, the minimal loss value of $\mathcal{L}_i$ is:

$$\mathcal{L}_i = -\log \frac{p_{sum}^+}{1+\alpha K_{y_i}} - \alpha K_{y_i} \log \frac{\alpha p_{sum}^+}{1+\alpha K_{y_i}}. \quad (17)$$

Thus, when $p_{sum}^+ = 1.0$, $\mathcal{L}_i$ achieves minimum with the optimal value $p_i^+ = \frac{\alpha}{1+\alpha K_{y_i}}$, which is the probability that two samples of the same class are a true positive pair, and the optimal value $p_c^+ = \frac{1}{1+\alpha K_{y_i}}$ which is the probability that a sample is closest to its corresponding center c_{y_i} among $\mathbf{C}$.

C. Gradient Derivation

In Section 3.4, we analyze PaCo loss under balanced setting, taking full ImageNet as an example. With P_{sup} increases from 0 to 0.71, the intensity of supervised contrastive loss will enlarge. Here we show that more samples will be pulled together with their corresponding centers when P_{sup} increases from 0 to 0.71 from the perspective of gradient derivation.

$$\frac{\partial \mathcal{L}}{\partial c_k} = \begin{cases} (\alpha K^* + 1) p_{c_k} x_i, & y_i \neq k; \\ \{(\alpha K^* + 1) p_{c_k} - 1\} \, x_i, & y_i = k. \end{cases} \tag{18}$$

It is worthy to note that when $p_{c_k} \in (0, 0.71)$, we have

$$\begin{cases} \dfrac{\partial \mathcal{L}}{\partial c_k} > 0, & y_i \neq k; \\ \dfrac{\partial \mathcal{L}}{\partial c_k} < 0, & y_i = k. \end{cases} \tag{19}$$

Eqs. (18) and (19) mean that as P_{sup} increases in training process, the probability that a sample is closest to its corresponding center will increase and the probability that a sample is closest to other centers will decrease. Thus, more and more samples will be pulled together with their right centers.

D. More Experimental Results on Many-shot, Medium-shot, and Few-shot.

Table 9: Comprehensive results on ImageNet-LT with different backbone networks (ResNet-50, ResNeXt-50 & ResNeXt-101). Models are trained with RandAugment in 400 epochs. Inference time is calculated with a batch of 64 images on Nvidia GeForce 2080Ti GPU, Pytorch1.5, Python3.6.

Backbone	Method	**Inference time (ms)**	Many	Medium	Few	All
ResNet-50	τ-normalize	8.3	65.0	52.2	32.3	54.5
	Balanced Softmax	8.3	66.7	52.9	33.0	55.0
	PaCo	8.3	65.0	55.7	38.2	57.0
ResNeXt-50	τ-normalize	13.1	66.4	53.4	38.2	56.0
	Balanced Softmax	13.1	67.7	53.8	34.2	56.2
	PaCo	13.1	67.5	56.9	36.7	58.2
ResNeXt-101	τ-normalize	25.0	69.0	55.1	36.9	57.9
	Balanced Softmax	25.0	69.2	55.8	36.3	58.0
	PaCo	25.0	68.2	58.7	41.0	60.0

Table 10: Comprehensive results on ImageNet-LT with RIDE. Models are trained with RandAugment in 400 epochs. Inference time is calculated with a batch of 64 images on Nvidia GeForce 2080Ti GPU, Pytorch1.5, Python3.6.

Backbone	Method	**Inference time (ms)**	Many	Medium	Few	All
RIDEResNet	1 expert	8.2	64.8	49.8	29.6	52.8
	2 experts	12.0	67.7	53.5	31.5	56.0
	3 experts	15.3	69.0	54.7	32.5	57.0
RIDEResNeXt	1 expert	13.0	67.2	49.0	28.1	53.2
	2 experts	19.0	70.4	52.6	30.3	56.4
	3 experts	26.0	71.8	53.9	32.0	57.8

Table 11: Comprehensive results on iNaturalist 2018 with ResNet-50 and ResNet-152. †represents the models are trained without RandAugment. Inference time is calculated with a batch of 64 images on Nvidia GeForce 2080Ti GPU, Pytorch1.5, Python3.6.

Backbone	Method	**Inference time (ms)**	Many	Medium	Few	All
ResNet-50	τ-normalize	8.3	74.1	72.1	70.4	71.5
	Balanced Softmax	8.3	72.3	72.6	71.7	71.8
	PaCo	8.3	70.3	73.2	73.6	73.2
ResNet-50 †	Balanced Softmax	8.3	72.5	72.3	71.4	71.7
	PaCo	8.3	69.5	73.4	73.0	73.0
ResNet-152	PaCo	20.1	75.0	75.5	74.7	75.2

Table 12: Comprehensive results on iNaturalist 2018 with RIDE. Models are trained with RandAugment in 400 epochs without knowledge distillation. Inference time is calculated with a batch of 64 images on Nvidia GeForce 2080Ti GPU, Pytorch1.5, Python3.6.

Backbone	Method	**Inference time (ms)**	Many	Medium	Few	All
RIDEResNet	1 expert	8.2	56.0	66.3	66.0	65.2
	2 experts	12.0	62.2	70.5	70.0	69.5
	3 experts	15.3	66.5	72.1	71.5	71.3

Table 13: Comparison with re-weighting baselines on ImageNet-LT with ResNet-50. The re-weighting strategy is applied to the supervised contrastive loss. Models are all trained without RandAugment.

Method	Top-1 Accuracy
CE	48.4
multi-task (CE+Re-weighting)	49.0
multi-task (CE+Blance Softmax)	48.6
PaCo	51.0

E. Implementation details for Table 1

We train models with cross-entropy, parametric contrastive loss 400 epochs without RandAugment respectively. For supervised contrastive loss, following the original paper, we firstly train the model 400 epochs. Then we fix the backbone and train a linear classifier 400 epochs.

F. Ablation Study

Re-weighting in contrastive learning without center learning rebalance Re-weighting is a classical method for dealing with imbalanced data. Here we directly apply the re-weighting method of Cui *et al.* [16] in contrastive learning to compare with PaCo. Moreover, Balanced softmax [38], as one state-of-the-art method for traditional cross-entropy in long-tailed recognition, is also applied to contrastive learning rebalance. The experimental results are summarized in Table 13. It is obvious PaCo significantly surpasses the two baselines.

Rebalance in center learning PaCo balances the contrastive learning (for moderating contrast among samples). However the center learning also needs to be balanced, which has been explored in [1, 25, 15, 20, 8, 41, 38, 28, 49, 14, 46, 18]. To compare with state-of-the-art methods in long-tailed recognition, we incorporate Balanced Softmax [38] into the center

Table 14: Comparison with re-weighting baselines that perform center learning rebalance on ImageNet-LT with ResNeXt-50. Models are all trained with RangAugment in 400 epochs.

Method	loss weight	Top-1 Accuracy
multi-task (Balanced Softmax+Re-weighting)	0.05	57.0
multi-task (Balanced Softmax+Re-weighting)	0.10	57.1
multi-task (Balanced Softmax+Re-weighting)	0.20	57.1
multi-task (Balanced Softmax+Re-weighting)	0.30	57.0
multi-task (Balanced Softmax+Re-weighting)	0.50	57.2
multi-task (Balanced Softmax+Re-weighting)	0.80	57.2
multi-task (Balanced Softmax+Re-weighting)	1.00	56.9
PaCo	-	58.2

learning. As shown in Table 14, after rebalance in center learning, PaCo boosts performance to 58.2%, surpassing baselines with a large margin.