

Learning Prototype Classifiers for Long-Tailed Recognition

Saurabh Sharma¹, Yongqin Xian², Ning Yu³ and Ambuj Singh¹

¹University of California Santa Barbara, USA

²Google, Switzerland

³Salesforce Research, USA

{saurabhsharma, ambuj}@cs.ucsb.edu, yxian@google.com, ning.yu@salesforce.com

Abstract

The problem of long-tailed recognition (LTR) has received attention in recent years due to the fundamental power-law distribution of objects in the real-world. Most recent works in LTR use softmax classifiers that are biased in that they correlate classifier norm with the amount of training data for a given class. In this work, we show that learning prototype classifiers addresses the biased softmax problem in LTR. Prototype classifiers can deliver promising results simply using Nearest-Class-Mean (NCM), a special case where prototypes are empirical centroids. We go one step further and propose to jointly *learn prototypes* by using distances to prototypes in representation space as the logit scores for classification. Further, we theoretically analyze the properties of Euclidean distance based prototype classifiers that lead to stable gradient-based optimization which is robust to outliers. To enable independent distance scales along each channel, we enhance Prototype classifiers by learning channel-dependent temperature parameters. Our analysis shows that prototypes learned by Prototype classifiers are better separated than empirical centroids. Results on four LTR benchmarks show that Prototype classifier outperforms or is comparable to state-of-the-art methods. Our code is made available at <https://github.com/saurabhsharma1993/prototype-classifier-ltr>.

1 Introduction

Imbalanced datasets are prevalent in the natural world due to the fundamental power-law distribution of objects [Van Horn and Perona, 2017]. Past decades have seen a lot of research in class-imbalanced learning [Estabrooks *et al.*, 2004; He and Garcia, 2009], and more recently, due to its far-reaching relevance in the era of deep-learning, the problem of long-tailed recognition (LTR) has received significant attention [Cui *et al.*, 2019; Cao *et al.*, 2019; Liu *et al.*, 2019] in the field of computer vision.

A common underlying assumption of recent work in LTR is that softmax classifiers learned from imbalanced train-

ing datasets are biased towards head classes. They seek to rectify head class bias in the classifier through data-resampling [Kang *et al.*, 2020], loss reshaping [Ren *et al.*, 2020; Menon *et al.*, 2021; Samuel and Chechik, 2021], ensemble-based models [Xiang *et al.*, 2020; Wang *et al.*, 2020; Zhou *et al.*, 2020], and knowledge transfer [Liu *et al.*, 2019; Liu *et al.*, 2021]. Particularly, it was shown in [Kang *et al.*, 2020] that deep models trained on imbalanced datasets have a high correlation of classifier norm to class size. To achieve similar classifier norms, they freeze the feature extractor and retrain the classifier with balanced class sampling in a second stage. Alternatively, they normalize every classifier by its L_2 norm or a power of its norm, leading to balanced classifier norms. More recently, [Alshammari *et al.*, 2022] showed that balanced weights in both the feature extractor and classifier can be achieved by regularizing the model with higher weight decay and/or MaxNorm constraints.

The shortcoming of correlating softmax classifier norm to class size is directly connected to the classification model. We show an illustration in Fig. 1. Consider the problem of learning a classifier f_W parameterized by weight matrix W . We assume the feature backbone g_θ is fixed and known. The classifier outputs logit scores $f_W(g(x)) = W^T g(x)$, and the posterior probability $p(y|g(x)) \propto p(y)p(g(x)|y) \propto (W^T g(x))_y$. Note that softmax classifier has both a magnitude and direction associated to it. In the ideal case, the directions get aligned with the class means μ_y to encode $p(g(x)|y)$ [Papyan *et al.*, 2020; Thrampoulidis *et al.*, 2022]. But for an imbalanced training dataset, to encode the prior of the label distribution $p(y)$ into classifier weights, the learned classifier norm gets correlated to the class size.

To circumvent this shortcoming of softmax in the long-tailed classification setting, it is advantageous to change the classification model from dot-product to distances in the representation space. More specifically, associate a learnable prototype c_y to each class y , and let $p(y|x) \propto -d(g(x), c_y)$, where $d(\cdot, \cdot)$ is a distance metric in the representation space. With a suitable choice of distance metric, such as Euclidean or cosine distances, we obtain a simple and effective prototype classifier which avoids the biased classifier norm problem. While prior work in long-tailed recognition explores the closely related Nearest-Class-Mean classifier [Kang *et al.*, 2020], which is a special case of our model with fixed prototypes, the full potential of prototype based classifiers has not

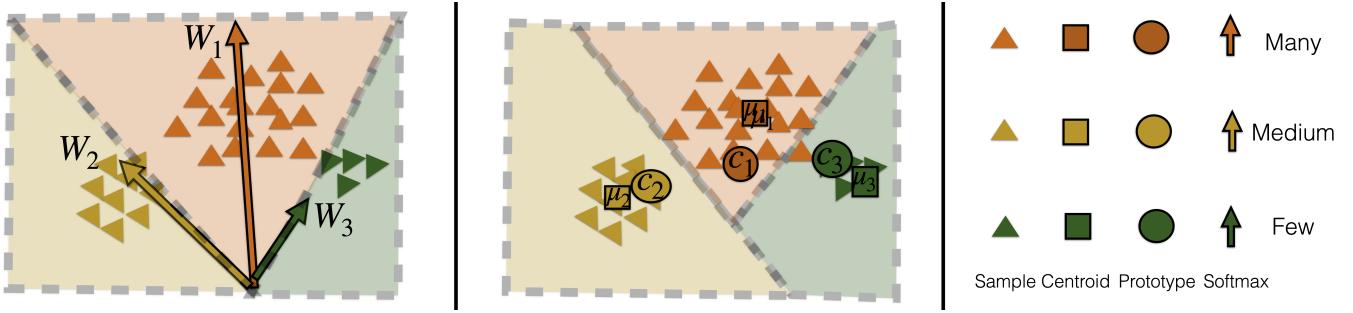


Figure 1: An illustration of Softmax vs Prototype classifiers for long-tailed data. Softmax classifiers have both a direction and a magnitude, indicated by the orientation and length of the classifier vector. During imbalanced training, the length of softmax classifiers gets correlated to the class size and leads to classification boundaries biased towards classes with Many samples. In contrast, prototype classifiers circumvent this shortcoming by using Euclidean distances to learnable prototypes within representation space, leading to more fair decision boundaries.

been fully explored. In this work we thoroughly analyze prototype classifiers and show that they offer a promising alternative to softmax in long-tailed recognition.

We theoretically analyze and show that the Euclidean distance based prototype classifier has stable gradient-based optimization properties for learning prototypes. In particular, the L2 norm of the gradient on a prototype c_y from a point x is always equal in magnitude to the misclassification probability. Further, the gradient is oriented along the line joining c_y to x . This leads to stable gradient updates that aren't affected by outliers that may be far from the prototypes.

We also allow for variable scales along the different dimensions of the feature representations in the prototype distance. This is enabled by learning channel dependent temperature parameters in the prototype distance function. We find that this is beneficial since it's a means of learnable feature scaling and/or selection in the prototype classifier. It also lets the model learn a general Mahalanobis distance metric parameterized by a diagonal matrix.

Overall, here is a summary of our main contributions:

(1) We propose a novel learnable prototype classifier for long-tailed recognition to counter the shortcoming of softmax to correlate classifier norm to class size. This offers a promising distance-based alternative to dot-product classifiers that is relatively underexplored in the long-tailed setting.

(2) We theoretically analyze the properties of Euclidean distance based prototype classifiers that leads to stable gradient-based optimization which is robust to outliers.

(3) We enhance prototype classifiers with channel dependent temperature parameters in the prototype distance function, allowing for learnable feature scaling and/or selection.

(4) We evaluate our proposed prototype classifier on four benchmark long-tailed datasets, CIFAR10-LT, CIFAR100-LT, ImageNet-LT and iNaturalist18. Our results show that the prototype classifier outperforms or is comparable to the recent state-of-the-art methods.

2 Related Work

Prototype-based classifiers. Prototype and Nearest-Class-Mean classifiers appear prominently in metric learning and few-shot learning. [Mensink *et al.*, 2013] proposed learning a Mahalanobis distance metric on top of fixed representations,

which leads to a NCM classifier based on the Euclidean distance in the learned metric space. [Snell *et al.*, 2017] learn deep representations for few shot learning by minimizing the prototype based loss on task-specific episodes. [Guerriero *et al.*, 2018] learn deep representations by optimizing the NCM classification objective on the entire training data. [Rebuffi *et al.*, 2017] learn NCM classifiers by maintaining a fixed number of samples to compute prototypes or centroids. In this work, we neither learn a distance metric or finetune representations using the NCM objective; instead our Prototype classifier directly learns the prototypes from pre-trained representations for long-tailed recognition.

Long-tailed recognition. The literature on long-tailed recognition is quite vast. Data resampling or reweighting is the most commonplace strategy against imbalanced datasets. This generally takes three forms: (i) Oversampling minority class samples by adding small perturbations to the data [Chawla *et al.*, 2002; Chawla *et al.*, 2003], (ii) Undersampling majority class samples by throwing away some data [Drummond *et al.*, 2003], and (iii) Uniform or class-balanced sampling based on the number of samples in a class [Sun *et al.*, 2019; Xian *et al.*, 2019]. However, oversampling minority class samples has been shown to lead to overfitting and undersampling majority class samples can cause poor generalization [He and Garcia, 2009]. Recently, [Kang *et al.*, 2020] learn the representations using instance-based sampling and retrain the softmax classifier in a second step using uniform sampling while keeping underlying representations fixed. We also use the two-stage training strategy in our work; however, we avoid learning an additional softmax classifier and instead use Prototype classifiers as it operates in the representation space directly. Another line of prior work focuses on engineering loss functions suited for imbalanced datasets. Focal loss [Lin *et al.*, 2017] reshapes the standard cross-entropy loss to push more weight on misclassified and/or tail class samples. Class-balanced loss [Cui *et al.*, 2019] uses a theoretical estimate of the volume occupied by a class to reweigh the loss function. [Cao *et al.*, 2019] offset the label's logit score with a power of class size to achieve a higher decision boundary margin for tail classes. [Ren *et al.*, 2020] apply the offset to all the logits, and change the class size exponent from 1/4 in prior work to 1 and use a

meta-sampler to retrain the classifier. [Menon *et al.*, 2021] apply a label-dependent adjustment to the logit score before softmax which is theoretically consistent. In our work, we build upon the logit adjustment approach and show how it can also be adapted to Prototype classifiers. Many other papers employ a specialized ensemble of experts to reduce tail bias and model variance. [Sharma *et al.*, 2020] train experts on class-balanced subsets of the training data and aggregate them using a joint confidence calibration layer. [Xiang *et al.*, 2020] also train experts on class-balanced subsets and distill them into a unified student model. [Wang *et al.*, 2020] explicitly enforce diversity in experts and aggregate them using a routing layer. [Zhou *et al.*, 2020] train two experts using regular and reversed sampling together with an adaptive learning strategy. In other works, self-supervised contrastive pretraining is used to improve feature representations for long-tailed recognition [Samuel and Chechik, 2021; Cui *et al.*, 2021]. More recently, [Alshammari *et al.*, 2022] show that quite good performance on long-tailed datasets is possible by simply tuning weight decay properly. Thus, we build our Prototype classifiers on top of the representation learning backbone provided by them. However, the gain of Prototype classifiers over Softmax is invariant to the choice of the backbone, as we show in our experiments.

3 Our Approach

3.1 Prototype Classifier

The Prototype classifier assigns probability scores to classes based on distances to learnable prototypes in representation space. We assume that the representations $g(x)$ are fixed. In other words, the weights θ of the feature extractor g_θ are learned in a prior stage and fixed to enable learning of prototype classifiers. The decoupling of representation and classifier learning is commonly used to train softmax classifiers in long-tailed recognition [Kang *et al.*, 2020]; here we adopt the same strategy to learn prototype classifiers.

Let the learnable prototype for class y be denoted by c_y . Under the assumption that the class-conditional distribution $p(g(x)|y)$ is a Gaussian with unit variance centered around the prototype c_y , we get the following probabilistic model for the class posterior $p(y|g(x))$:

$$\begin{aligned} \log p(y|g(x)) &\propto -\frac{1}{2}d(g(x), c_y) \\ &= \log \frac{e^{-\frac{1}{2}d(g(x), c_y)}}{\sum_{y'} e^{-\frac{1}{2}d(g(x), c'_y)}} \end{aligned} \quad (1)$$

This leads to the nearest class prototype decision function for classification:

$$y^* = \operatorname{argmin}_{y \in \{1, \dots, \bar{y}\}} d(g(x), c_y)$$

Different choices of distance metric lead to different prototype classifiers. While squared Euclidean distance and Euclidean distance lead to equivalent decision boundaries, we find that Euclidean distance is more amenable for learning of the prototypes due to its stable gradient. The Prototype classifier with Euclidean distance metric specifically uses:

$$d(g(x), y) = \sqrt{(g(x) - c_y)^T (g(x) - c_y)} \quad (2)$$

The prototypes are learned using the Maximum Likelihood Estimation principle by minimizing the Negative Log Likelihood objective function:

$$L = -\frac{1}{N} \sum_i^N \log p(y_i|g(x_i)) \quad (3)$$

The remainder of this section is organized as follows: in Section 3.2, we show how Euclidean distance leads to stable gradient updates on prototypes, then in Section 3.3 we show how learning prototypes addresses the biased softmax problem, followed by Section 3.4 which details our channel-dependent temperatures to enhance prototype learning, and Section 3.5 which describes logit adjustment for learning prototypes. Our entire pipeline is presented in Algorithm 1.

3.2 Stable Gradient Updates on Prototypes

In Prototype classifier, we use the Euclidean distance in Eq. 2 due to its stable gradient optimization. We now describe the gradient updates on the prototypes.

Theorem 1. *Let L_{xy} denote the NLL loss on a single sample x with label y . Then, the L_2 norm of the gradient on prototype c_z is given by*

$$\left\| \frac{\partial L_{xy}}{\partial c_z} \right\|_2 = \begin{cases} 1 - p(y|g(x)) & : z = y \\ p(z|g(x)) & : z \neq y \end{cases}$$

Further, if $z = y$, then the gradient descent direction lies along $g(x) - c_z$, else it is exactly the opposite direction.

Proof. Let $Z(x) = \sum_{y'} e^{-\frac{1}{2}d(g(x), c'_y)}$ denote the normalization term over all the classes, and δ_{yz} denotes the Kronecker delta. Then, we get a closed form expression for the gradient using repeated application of the chain rule,

$$\begin{aligned} \frac{\partial L_{xy}}{\partial c_z} &= \frac{\partial (-\log \frac{e^{-\frac{1}{2}d(g(x), c_y)}}{Z(x)})}{\partial c_z} \\ &= \frac{\partial (-\log \frac{e^{-\frac{1}{2}\sqrt{c_y^T c_y - 2g(x)^T c_y + g(x)^T g(x)}}}{Z(x)})}{\partial c_z} \\ &= \frac{\delta_{yz}(1 - p(z|g(x)))(c_z - g(x))}{d(g(x), c_z)} \\ &\quad + \frac{(1 - \delta_{yz})p(z|g(x))(g(x) - c_z)}{d(g(x), c_z)} \end{aligned}$$

□

From the gradient update, we see that if sample x belongs to class y , then the updated prototype is shifted in the direction $(g(x) - c_z)$, weighted by the misclassification probability $(1 - p(z|x))$, otherwise it's shifted in the opposite direction weighted by the misclassification probability $p(z|x)$.

In both cases, the L_2 norm of the gradient is equal to the misclassification probability, leading to stable gradient updates. Importantly, it is invariant to $d(g(x), c_z)$, thus both inliers and outliers contribute equally. On the other hand, if we use squared Euclidean distance, it is easy to see that the L_2 norm of the gradient is scaled by $d(g(x), c_z)$, which is highly sensitive to outliers. We also verify empirically in Section 4.4 that Euclidean distance outperforms squared Euclidean distance for learning prototypes.

3.3 Addressing the Biased Softmax Problem

We show how learning prototypes addresses the biased softmax problem by drawing a connection with the softmax classifier. We consider below a Prototype classifier based on the squared Euclidean distance metric:

$$\begin{aligned} -\frac{1}{2}d(g(x), c_y) &= -\frac{1}{2}(g(x) - c_y)^T(g(x) - c_y) \\ &= -\frac{1}{2}(c_y^T c_y - 2g(x)^T c_y + g(x)^T g(x)) \end{aligned}$$

Since the term $-\frac{1}{2}g(x)^T g(x)$ is shared by all classes, it can be ignored. Thus, the Prototype classifier with squared Euclidean distance metric is a softmax classifier with weight c_y and bias $-\frac{1}{2}c_y^T c_y$. Now, we know that softmax can increase or decrease the norm of its weight terms to account for an imbalanced prior label distribution $p(y)$ [Kang *et al.*, 2020]. However, the same is not true for the Prototype classifier, because the bias term negates the gains from increasing or decreasing the norm of the weight term. Due to this coupling of the weight and bias terms, the Prototype classifier is more robust to imbalanced distributions and learns equinorm prototypes. We further show in our experiments in Section 4.5 that the learned prototypes are actually very similar in norm.

3.4 Channel-Dependent Temperatures

While computing distances for prototype classifiers, it is worthwhile to consider that all dimensions in the representation space may not have the same distance scale. Specifically, for a large distance along a dimension with high variance may not mean as much a large distance along a dimension with low variance. Thus, we define the channel-dependent temperature (CDT) prototype distance function using learnable temperatures T_i ,

$$d_{CDT}(g(x), y) = \sqrt{\sum_{i=1}^d \frac{(g(x) - c_y)_i^2}{T_i}} \quad (4)$$

A high value of temperature T implies low sensitivity to distances along that dimension, and a low value implies high sensitivity. Learnable temperatures allow the model to learn feature scaling in the distance function. By letting the temperature approach very high values the model can also filter out potentially harmful features, i.e., feature selection. Finally, the CDT distance can also be interpreted as a Mahalanobis distance $\sqrt{(g(x) - c_y)^T \Sigma^{-1} (g(x) - c_y)}$, where Σ is a diagonal matrix with the temperatures on the diagonal.

3.5 Logit Adjustment for Learning Prototypes

Recent work by [Menon *et al.*, 2021] suggests that, for a classifier trained on an imbalanced dataset, it is important to perform an additive adjustment to the logits based on the distribution prior. Motivated by this strategy, we modify the prototype classifier loss function in Eq. 3 as follows:

$$L_{xy} = -\log \frac{e^{-\frac{1}{2}d(g(x), c_y) + \tau \cdot \log N_y}}{\sum_{y'} e^{-\frac{1}{2}d(g(x), c_{y'}) + \tau \cdot \log N_{y'}}} \quad (5)$$

Algorithm 1: Learning prototype classifier

Require: Pretrained feat reps $g(x)$, class means μ_y
Require: Class-sizes N_y , logit-adj weight τ , lrs α, β
 Initialize prototypes $c_y = \mu_y$ and temperatures $T_i = 1$;
for $j \leftarrow 0$ **to** $num_batches$ **do**
 Sample class-balanced batch $(g(x_k), y_k)_{k=1}^{batchsize}$;
 $\forall k \& \forall y'$ compute $d_{CDT}(g(x_k), c_{y'})$ using Eq. 4;
 Compute $L = \sum_k L_{x_k y_k}$ using Eq. 5;
 $\forall y'$ update $c_{y'} \leftarrow c_{y'} - \alpha \nabla_{c_{y'}} L$;
 $\forall T_i$ update $T_i \leftarrow T_i - \beta \nabla_{T_i} L$;

Here N_y denotes the number of training samples for class y and τ is a hyperparameter. Logit adjustment is only done during training and not during inference. It places a higher penalty on misclassifying tail class samples relative to head classes. Empirical results show that logit adjustment does yield better results for learning prototype classifiers.

4 Experiments

4.1 Datasets

We evaluate our proposed method on the following three benchmark long-tailed datasets: 1. **CIFAR100-LT** [Cao *et al.*, 2019]: CIFAR100 consists of 60K images from 100 classes. Following prior work, we exponentially decay number of training samples and control the degree of data imbalance with an imbalance factor β . $\beta = N_{max}/N_{min}$, where N_{max} and N_{min} are the maximum and minimum number of training images per class respectively. N_{max} is kept fixed at 500, and we report on three commonly used imbalance ratios $\beta \in [100, 50, 10]$. The validation set consists of 100 images per class and is also used as the test set. 2. **ImageNet-LT** [Liu *et al.*, 2019]: This is a long-tailed split of ImageNet. ImageNet-LT has an imbalanced training set with 115,846 images for 1,000 classes from ImageNet-1K [Deng *et al.*, 2009]. The class frequencies follow a natural power-law distribution [Van Horn and Perona, 2017] with a maximum of 1,280 and a minimum of 5 images per class. The validation and testing sets are balanced and contain 20 and 50 images per class respectively. 3. **iNaturalist-LT** [Van Horn *et al.*, 2018]: This is a species classification dataset that naturally follows a long-tailed distribution. There are 8,142 classes, with a maximum of 118,800 images per class and a minimum of 16. The validation set contains 3 images per class and is also used as the test set.

4.2 Evaluation Protocol

To distinguish the performance of the model on head vs tail classes, we report average top-1 accuracy on balanced test sets across four splits, *Many*: classes with ≥ 100 samples, *Med*: classes with $20 \sim 100$ samples, *Few*: classes < 20 samples, and *All* classes. Since the test sets are balanced, average accuracy and mean precision are the same.

4.3 Implementation Details

We train prototype classifiers on fixed representations from a prior pre-trained model. Our training pipeline consists of two

stages, representation learning followed by prototype learning. To compare fairly with prior work, we use the same backbone architectures: ResNet-32 for CIFAR100, ResNeXt50 for ImageNet-LT and ResNet-50 for iNaturalist18.

Representation learning. We train all models end to end with an auxilliary softmax classifier by minimizing the average cross-entropy classification loss. At the end of training we discard the softmax classifier. We use instance-balanced data sampling [Kang *et al.*, 2020] and regularize model weights using high weight decay [Alshammari *et al.*, 2022], which achieves balanced weights in the feature extractor. We use the SGD optimizer with momentum 0.9, learning rate 0.01 along with cosine learning rate schedule [Loshchilov and Hutter, 2017] decaying to 0 for all models. For CIFAR100-LT and ImageNet-LT we train for 200 epochs with batch size 64 and for iNaturalist18 we train for 200 epochs with batch size 512. The weight decay parameter is found using Bayesian hyperparameter search [Nogueira and others, 2014]. We use random cropping and horizontal flipping for training data augmentation. All our training is done on NVIDIA V100 GPUs.

Prototype learning. We freeze representations and compute empirical centroids to initialize the class prototypes. We minimize the average logit-adjusted cross entropy loss with logit-adjustment weight $\tau = 1/4$. We train for 1 epoch with class-balanced sampling, set prototype learning rate 4, CDT learning rate 0.005 and use the SGD optimizer with momentum 0.9. We also use random cropping, horizontal flipping, Autoaug [Cubuk *et al.*, 2018] and Cutout [DeVries and Taylor, 2017] for data augmentation.

4.4 Ablation Study

Model components. The benefits gained by adding different components to our Prototype classifier (PC) are investigated in an ablation study over CIFAR100-LT with imbalance ratio 100, shown in Table 1. We make several observations: (i) NCM, which is a special case of our model where the empirical centroids are used as class prototypes, i.e $c_y = \frac{1}{N_i}g(x)$, makes considerable gains above softmax, boosting *All* accuracy from 46.12 to 48.67 and *Few* accuracy from 11.34 to 14.13. NCM does not suffer from classifier norm correlation to class size like softmax and can do better even without any learning. (ii) Prototype classifier (PC) lifts us above NCM significantly, achieving 4% improvement on *All* accuracy and 10% improvement on *Few* accuracy. Thus, learning prototypes gives a far superior nearest-prototype model than simply using the class-centroids as done by previous work. (iii) Adding channel dependent temperatures (CDT) to PC further improves the results by 0.5% on *All* accuracy and 4% on *Few* accuracy. CDT enhances the model by effecting distinct distance scales for all the features. (iv) Logit adjustment (LA) further improves results for PC. By combining both CDT and LA to PC we’re able to gain 1% and 9% improvements over the baseline PC in *All* and *Few* accuracies respectively. In the remainder, PC refers to our best model PC + CDT + LA.

Distance metrics. We experiment with different distance metrics for the Prototype classifier, particularly (a) Euclidean

Method	Many	Med	Few	All
Softmax	76.12	46.13	11.34	46.12
NCM	76.54	50.40	14.13	48.67
PC	74.00	55.17	24.50	52.56
+ CDT	70.77	55.29	28.63	53.06
+ LA	69.29	54.63	32.67	53.17
+ CDT + LA	70.17	55.89	33.93	53.41

Table 1: Ablation study for CIFAR100-LT with imbalance ratio 100. NCM, PC, CDT and LA refer to Nearest-Class-Mean, Prototype classifier, Channel-dependent Temperature and Logit adjustment respectively. NCM alone with class centroids outperforms Softmax. PC via learning prototypes outperforms NCM by 4% on *All* accuracy. Adding CDT and LA further yields 1% improvement in *All* accuracy.

Method	Many	Med	Few	All
Softmax	76.12	46.13	11.34	46.12
PC + Euclidean	70.17	55.89	33.93	53.41
PC + Squared Euclidean	39.89	21.77	8.10	24.01
PC + Cosine	72.94	54.80	24.77	52.14

Table 2: Effect of different distance metrics in learning prototypes on CIFAR100-LT. Euclidean distance outperforms Squared Euclidean and Cosine distance. Note the really bad performance of Squared Euclidean, which agrees with our theoretical analysis that its prone to outliers.

distance from Eq. 2, (b) Squared Euclidean distance and (c) Cosine distance: $d(g(x), c_y) = 1 - \frac{g(x)^T c_y}{\|g(x)\| \|c_y\|}$. The results on CIFAR100-LT are depicted in Table 2. We make some important observations: (i) Squared euclidean distance does considerably worse, with *All* accuracy dropping by more than 30%. This agrees with our theoretical analysis that the Squared euclidean distance is highly sensitive to outliers for learning prototypes. (ii) Cosine distance underperforms Euclidean distance on *All* and *Tail* accuracy. This suggests that the geometry of the representation space is closer to Euclidean and not hyperspherical.

Feature backbones. Since Prototype classifier is agnostic to the feature backbone used to generate representations, we experiment with various representation learning and classifier learning schemes from the LTR literature. Particularly, we use CE, DRO-LT [Samuel and Chechik, 2021] and WD [Alshammari *et al.*, 2022] as pre-trained feature backbones. For classifier learning, we use CE, cRT [Kang *et al.*, 2020], WD + MaxNorm [Alshammari *et al.*, 2022] and Prototype classifier (PC). The results on CIFAR100-LT are depicted in Table 3. We observe that Prototype classifier outperforms the competing classifier learning schemes for all the different feature backbones, thus showing that the performance gain is agnostic to the choice of backbone. Our best results are obtained using the WD baseline, which applies strong weight decay for regularizing feature weights.

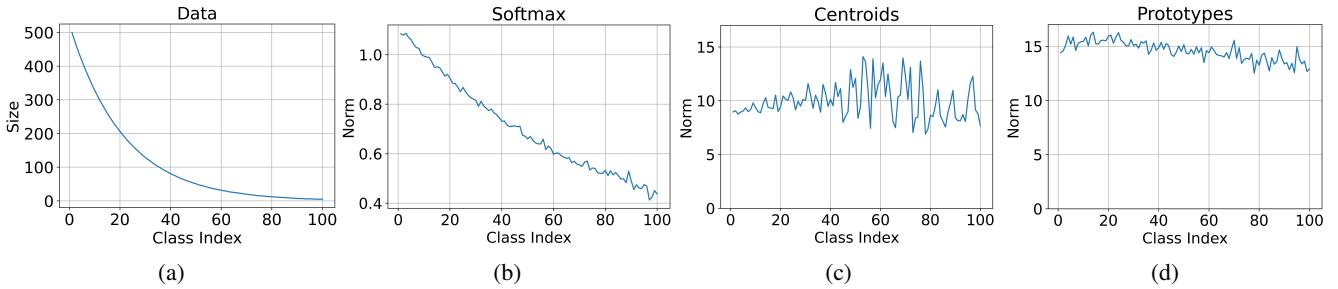


Figure 2: Comparison of softmax and prototype norms. L-R: a) Class size vs index, b),c) and d): Norm of softmax classifier, centroids and prototypes vs class index respectively on CIFAR100-LT. Classes are sorted according to decaying class size. Prototype classifier achieves balanced norms, in contrast to the biased softmax classifier and the imbalanced centroid norms.

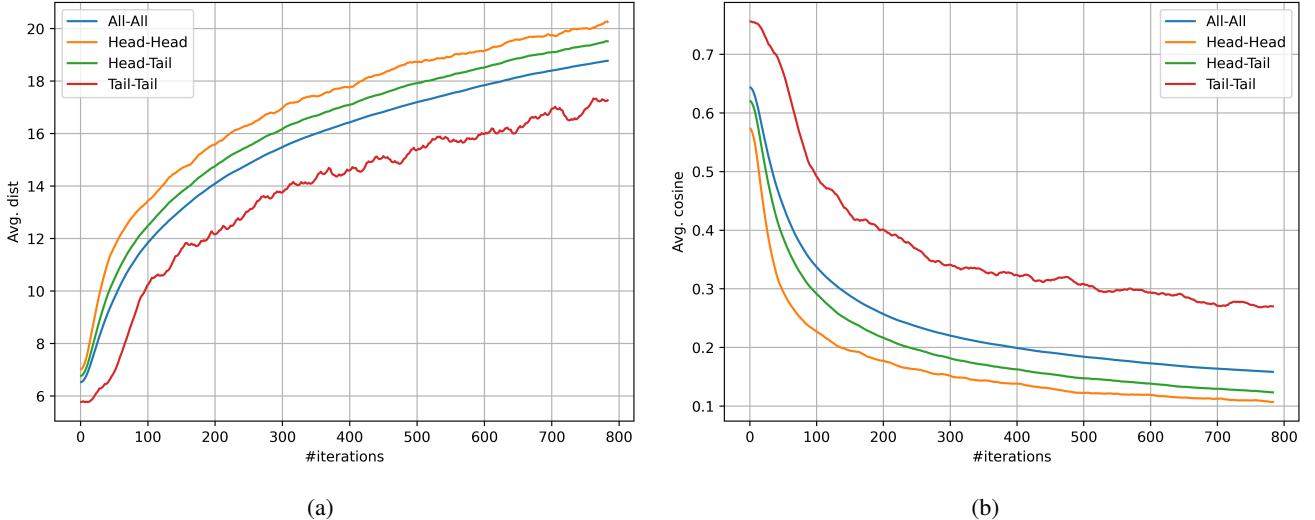


Figure 3: Evolution of prototypes during training. a) The average Euclidean distance between prototypes vs number of training iterations. b) The average cosine similarity between prototypes vs number of training iterations, a lower value implies higher angular separation. We compare across pairs of All-All, Head-Head, Head-Tail and Tail-Tail class prototypes. Euclidean and angular separation increase monotonically. Tail-Tail classes have smaller separation than Head-Head, which is due to the Minority Collapse effect in imbalanced training.

		Classifier learning			
Representation Learning	CE	cRT	WD + MaxNorm	PC (Ours)	
CE	38.3	41.3	43.4	44.1	
DRO-LT	41.2	47.3	47.8	48.2	
WD	46.1	50.1	53.3	53.4	

Table 3: Effect of different representation and classifier learning schemes for CIFAR100-LT. PC outperforms competing classifier learning schemes for all different backbones.

4.5 Prototype Norms

To understand the advantage of prototype classifier over softmax, we plot the norm of softmax classifier, empirical centroids and learned prototypes In Fig. 2. We recover the correlation of classifier norm to class size for softmax, as shown by prior work. For empirical centroids, it is interesting to see that this correlation does not hold; however the centroid

norms are far from balanced. After learning the prototypes, we see that the norms become more balanced, with a slight favor towards the head classes. Since higher norm implies a negative bias in the prototype classifier, this actually favors the tail classes slightly.

4.6 How Do Prototypes Evolve During Training?

We are interested in the evolution of the class prototypes as training proceeds. Particularly, we seek to know the average separation in terms of Euclidean distance, as well as the average cosine similarity between the class prototypes. In Fig. 3, we plot the average of these values between pairs of All-All classes, Head-Head classes, Head-Tail classes and Tail-Tail classes. The average distance increases monotonically for all the pairs. The separation between Head-Head classes is greater than Tail-Tail classes. Similarly, the average cosine similarity decreases monotonically for all the pairs, therefore the angular separation increases. There is higher angular separation between Head-head classes over Tail-Tail classes. Thus, the prototype classifier learns prototypes that are better

separated than the empirical centroids, leading to lower classification error using the nearest-prototype decision function. It also mitigates Minority Collapse, wherein Minority Class means collapse towards the same point [Fang *et al.*, 2021].

5 Comparison to State-of-the-Art

Compared methods. We compare against recently published and relevant methods in the LTR field. We choose representative models from the different solutions spaces- (i) data reweighting: Classifier retraining (cRT) [Kang *et al.*, 2020] and DisAlign [Zhang *et al.*, 2021], (ii) loss reshaping: Focal loss [Lin *et al.*, 2017], Label-distribution aware margin (LDAM) loss [Cao *et al.*, 2019] and Logit-adjustment loss [Menon *et al.*, 2021], (iii) Ensemble-based models: RIDE [Wang *et al.*, 2020], (iv) Self-supervised contrastive learning: DRO-LT [Samuel and Chechik, 2021] and PaCO [Cui *et al.*, 2021], and (v) Weight Regularization: WD, and WD + WD + Max [Alshammary *et al.*, 2022].

Results. Our results on CIFAR100-LT, ImageNet-LT and iNaturalist18 are reported in Table 4, Table 5 and Table 6 respectively. The numbers of compared methods are taken from their respective papers. Prototype classifier outperforms the prior state-of-the-art methods DRO-LT, DisAlign etc. on all imbalance ratios in CIFAR100. We outperform WD + WD + Max, with whom we share the weight decay tuned baseline WD, on all three benchmark datasets. For ImageNet-LT, Prototype classifier outperforms all prior state-of-the-arts on *Tail* and *All* accuracy except models with “bells and whistles,” particularly RIDE, which learns and fuses multiple models into an ensemble. A similar conclusion can be derived on iNaturalist18, where our results rival prior state-of-the-arts, except the ensemble model RIDE, and PaCO, which uses self-supervised pre-training and aggressive data augmentation techniques [Cubuk *et al.*, 2020; He *et al.*, 2020]. However, since Prototype classifier is a classifier layer that can be applied to any feature backbone, our method is complimentary to ensembles and self-supervised pre-training. We leave this exploration for future work.

6 Conclusion

In this article, we proposed Prototype classifiers to address the biased softmax softmax problem for long-tailed recognition. Prototype classifiers do not suffer from the biased softmax problem of correlating classifier norm to class size. This is because they rely on distances to learnable prototypes in representation space instead of dot-product with learnable weights, which leads to a coupling of the weight and bias terms. Our theoretical analysis shows that Euclidean distance is stable for gradient-based optimization, and this is confirmed empirically. We further enable independent distance scales in the prototype distance function using channel-dependent temperatures. Our results on four benchmark long-tailed datasets show that Prototype classifier outperforms or is comparable to the prior state-of-the-art. Moreover, experiments reveal that the learned prototypes are equinorm and are much better separated in representation space than centroids..

Imbalance ratio	100	50	10
CE	38.32	43.85	55.71
Focal Loss	38.41	44.32	55.78
LDAM Loss	39.60	44.97	56.91
cRT	41.24	46.83	57.93
LogitAdjust	42.01	47.03	57.74
DRO-LT	47.31	57.57	63.41
PaCO	52.00	56.00	64.20
RIDE	49.10	-	-
WD	46.08	52.71	66.03
WD + WD + Max	<u>53.35</u>	<u>57.71</u>	<u>68.67</u>
NCM	48.67	53.62	67.82
PC (Ours)	53.41	57.75	69.12

Table 4: Comparison to state-of-the-art on CIFAR100-LT. Prototype classifier achieves superior results across all imbalance ratios.

	Many	Med	Few	All
CE	<u>65.9</u>	37.5	7.7	44.4
Focal Loss	36.41	29.9	16.0	30.5
cRT	61.8	46.2	27.3	49.6
DRO-LT	64.0	49.8	33.1	53.5
DisAlign	61.3	52.2	31.4	52.9
WD	68.5	42.4	14.2	48.6
WD + WD + Max	62.5	50.4	<u>41.5</u>	<u>53.9</u>
NCM	62.0	48.7	27.6	50.1
PC (Ours)	63.5	<u>50.8</u>	42.7	54.9
SOTA with “bells and whistles”				
RIDE	67.9	52.3	36.0	56.1
PaCO	63.2	51.6	39.2	54.4

Table 5: Comparison to state-of-the-art on ImageNet-LT. We outperform all prior state-of-the-art with single models and no self-supervised pretraining or aggressive data augmentation.

	Many	Med	Few	All
CE	<u>72.2</u>	63.0	57.2	61.7
Focal Loss	-	-	-	61.1
cRT	69.0	66.0	63.2	65.2
DRO-LT	-	-	-	69.7
DisAlign	69.0	71.1	70.2	70.6
WD	74.5	66.5	61.5	65.4
WD + WD + Max	71.2	70.4	<u>69.7</u>	<u>70.2</u>
NCM	61.0	63.5	63.3	63.1
PC (Ours)	71.6	<u>70.6</u>	70.2	70.6
SOTA with “bells and whistles”				
RIDE	66.5	72.1	71.5	71.3
PaCO	69.5	72.3	73.1	72.3

Table 6: Comparison to state-of-the-art on iNaturalist18. We outperform all prior state-of-the-art with single models and no self-supervised pretraining or aggressive data augmentation.

Acknowledgements

Research was partially sponsored by the NSF under award #2229876.

References

[Alshammari *et al.*, 2022] Shaden Alshammari, Yu-Xiong Wang, Deva Ramanan, and Shu Kong. Long-tailed recognition via weight balancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6897–6907, 2022.

[Cao *et al.*, 2019] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. In *NeurIPS*, 2019.

[Chawla *et al.*, 2002] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *JAIR*, 2002.

[Chawla *et al.*, 2003] Nitesh V Chawla, Aleksandar Lazarevic, Lawrence O Hall, and Kevin W Bowyer. Smoteboost: Improving prediction of the minority class in boosting. In *European conference on principles of data mining and knowledge discovery*, 2003.

[Cubuk *et al.*, 2018] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.

[Cubuk *et al.*, 2020] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020.

[Cui *et al.*, 2019] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *CVPR*, 2019.

[Cui *et al.*, 2021] Jiequan Cui, Zhisheng Zhong, Shu Liu, Bei Yu, and Jiaya Jia. Parametric contrastive learning. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 715–724, 2021.

[Deng *et al.*, 2009] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[DeVries and Taylor, 2017] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[Drummond *et al.*, 2003] Chris Drummond, Robert C Holte, et al. C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In *Workshop on learning from imbalanced datasets II*, 2003.

[Estabrooks *et al.*, 2004] Andrew Estabrooks, Taeho Jo, and Nathalie Japkowicz. A multiple resampling method for learning from imbalanced data sets. *Computational intelligence*, 2004.

[Fang *et al.*, 2021] Cong Fang, Hangfeng He, Qi Long, and Weijie J Su. Exploring deep neural networks via layer-peeled model: Minority collapse in imbalanced training. *Proceedings of the National Academy of Sciences*, 118(43):e2103091118, 2021.

[Guerriero *et al.*, 2018] Samantha Guerriero, Barbara Caputo, and Thomas Mensink. Deepncm: Deep nearest class mean classifiers. In *International Conference on Learning Representations Workshop*, 2018.

[He and Garcia, 2009] Haibo He and Edwardo A Garcia. Learning from imbalanced data. *TKDE*, 2009.

[He *et al.*, 2020] 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.

[Kang *et al.*, 2020] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. In *Eighth International Conference on Learning Representations (ICLR)*, 2020.

[Lin *et al.*, 2017] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017.

[Liu *et al.*, 2019] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *CVPR*, 2019.

[Liu *et al.*, 2021] Bo Liu, Haoxiang Li, Hao Kang, Gang Hua, and Nuno Vasconcelos. Gistnet: a geometric structure transfer network for long-tailed recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8209–8218, 2021.

[Loshchilov and Hutter, 2017] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017.

[Menon *et al.*, 2021] Aditya Krishna Menon, Sadeep Jayasumana, Ankit Singh Rawat, Himanshu Jain, Andreas Veit, and Sanjiv Kumar. Long-tail learning via logit adjustment. In *International Conference on Learning Representations*, 2021.

[Mensink *et al.*, 2013] Thomas Mensink, Jakob Verbeek, Florent Perronnin, and Gabriela Csurka. Distance-based image classification: Generalizing to new classes at near-zero cost. *TPAMI*, 2013.

[Nogueira and others, 2014] Fernando Nogueira et al. Bayesian optimization: Open source constrained global optimization tool for python. *URL https://github.com/fmfn/BayesianOptimization*, 2014.

[Papyan *et al.*, 2020] Vardan Papyan, XY Han, and David L Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117(40):24652–24663, 2020.

[Rebuffi *et al.*, 2017] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl:

Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017.

[Ren *et al.*, 2020] Jiawei Ren, Cunjun Yu, Xiao Ma, Haiyu Zhao, Shuai Yi, et al. Balanced meta-softmax for long-tailed visual recognition. *Advances in Neural Information Processing Systems*, 33:4175–4186, 2020.

[Samuel and Chechik, 2021] Dvir Samuel and Gal Chechik. Distributional robustness loss for long-tail learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[Sharma *et al.*, 2020] Saurabh Sharma, Ning Yu, Mario Fritz, and Bernt Schiele. Long-tailed recognition using class-balanced experts. In *DAGM German Conference on Pattern Recognition*. Springer, 2020.

[Snell *et al.*, 2017] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *NeurIPS*, 2017.

[Sun *et al.*, 2019] Qianru Sun, Yaoyao Liu, Tat-Seng Chua, and Bernt Schiele. Meta-transfer learning for few-shot learning. In *CVPR*, 2019.

[Thrampoulidis *et al.*, 2022] Christos Thrampoulidis, Ganesh Ramachandra Kini, Vala Vakilian, and Tina Behnia. Imbalance trouble: Revisiting neural-collapse geometry. In *Advances in Neural Information Processing Systems*, 2022.

[Van Horn and Perona, 2017] Grant Van Horn and Pietro Perona. The devil is in the tails: Fine-grained classification in the wild. *arXiv preprint arXiv:1709.01450*, 2017.

[Van Horn *et al.*, 2018] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist species classification and detection dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8769–8778, 2018.

[Wang *et al.*, 2020] Xudong Wang, Long Lian, Zhongqi Miao, Ziwei Liu, and Stella Yu. Long-tailed recognition by routing diverse distribution-aware experts. In *International Conference on Learning Representations*, 2020.

[Xian *et al.*, 2019] Yongqin Xian, Saurabh Sharma, Bernt Schiele, and Zeynep Akata. f-vaegan-d2: A feature generating framework for any-shot learning. In *CVPR*, 2019.

[Xiang *et al.*, 2020] Liuyu Xiang, Guiguang Ding, and Jun-gong Han. Learning from multiple experts: Self-paced knowledge distillation for long-tailed classification. In *European Conference on Computer Vision*. Springer, 2020.

[Zhang *et al.*, 2021] Songyang Zhang, Zeming Li, Shipeng Yan, Xuming He, and Jian Sun. Distribution alignment: A unified framework for long-tail visual recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2361–2370, 2021.

[Zhou *et al.*, 2020] Boyan Zhou, Quan Cui, Xiu-Shen Wei, and Zhao-Min Chen. Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9719–9728, 2020.

Appendix

A Dataset statistics

We detail the dataset statistics for the three benchmark long-tailed recognition datasets in Table 7.

Dataset	Attribute	Many	Medium	Few	All
CIFAR10-LT (Imba 100)	Classes	8	2	0	10
	Samples	12273	133	0	12406
CIFAR100-LT (Imba 100)	Classes	35	35	30	100
	Samples	8824	1718	305	10847
CIFAR100-LT (Imba 50)	Classes	41	40	19	100
	Samples	10331	2008	269	12608
CIFAR100-LT (Imba 10)	Classes	70	30	0	100
	Samples	17743	2130	0	19573
ImageNet-LT	Classes	391	473	136	1,000
	Samples	89,293	24,910	1,643	115,846
iNaturalist18	Classes	842	4076	3599	8,142
	Samples	433,806	258,340	133,061	437,513

Table 7: Statistics for training data in CIFAR10-LT, CIFAR100-LT, ImageNet-LT and iNaturalist18.

B More training details

We mention here a few more pertinent training details. The prototype learning is most optimal with high prototype learning rate. The effect of the prototype learning rate is shown in Fig. 4a. Further, we find that training for 1 epoch is enough. The effect of training epochs is shown in Fig. 4b. Training beyond 1 epoch leads to lower validation accuracy. Thus, the prototype learning has very fast convergence and is quick to train.

C Temperature schemes

We discuss here some other temperature schemes for learning prototypes. Apart from Channel-dependent temperatures, we also experiment with Class-dependent temperatures. We vary the temperature according to class, allowing for different distance scales specific to the local neighborhood of the class. Specifically, we only allow a per-class temperature T_y :

$$d_{Class}(g(x), y) = \sqrt{\frac{\sum_{i=1}^d (g(x) - c_y)_i^2}{T_y}} \quad (6)$$

Secondly, we also experiment with Dense temperatures, allowing different temperature scales per-class as well as per-channel T_{yi} :

$$d_{Dense}(g(x), y) = \sqrt{\sum_{i=1}^d \frac{(g(x) - c_y)_i^2}{T_{yi}}} \quad (7)$$

The three schemes, Channel temps, Class temps and Dense temps are compared in Table 8. We find that both Class temps and Dense temps have similar results and underperform Channel temps. We hypothesize this is because Channel-temps encourage knowledge transfer from Many to Few shot classes by enabling globally shared feature selection.

Method	Many	Med	Few	All
PC	74.00	55.17	24.50	52.56
+ Channel T	70.77	55.29	28.63	53.06
+ Class T	72.33	54.83	25.62	52.47
+ Dense T	71.28	54.77	24.11	52.17

Table 8: Effect of different temperature schemes learning prototypes, i.e Channel-dependent Temperature, Class-Dependent and Dense temperatures respectively, on CIFAR100-LT. Channel-dependent does the best.

D Effect of logit adjustment weight

In Fig. 4c, we study the effect of the logit adjustment weight on the different performance metrics. We observe a clear trade-off between *Few* accuracy on one hand and *All*, *Many* and *Med* on the other, with higher weight favoring *Few*. Consider that the logit adjustment $\log N_y$ leads to a higher loss on tail classes during the optimization process. Therefore, increasing the weight parameter leads to tail loss being over-emphasized during training and causing lower tail class error at the price of higher head class error. We choose $\tau = 1/4$ as the optimal weight achieving good accuracy across the spectrum.

E Results on CIFAR10-LT

The results for CIFAR10-LT with imbalance ratio 100 are depicted in Table 9. The results are similar to CIFAR100-LT, we surpass the competing state-of-the-art methods cRT [Kang *et al.*, 2020], LDAM [Cao *et al.*, 2019], DRO-LT [Samuel and Chechik, 2021], and WD + WD + Max [Alshamari *et al.*, 2022].

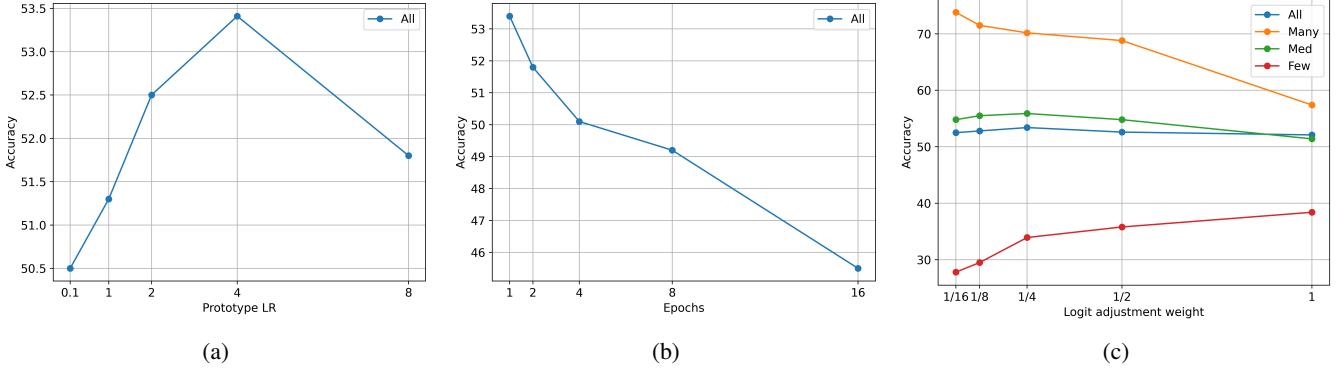


Figure 4: L-R: a) Effect of the prototype learning rate on CIFAR100-LT. b) Effect of the number of training epochs on CIFAR100-LT. c) Effect of the logit-adjustment weight τ on CIFAR100-LT.

All	
CE	72.1
cRT	73.5
LDAM	77.03
DRO-LT	82.6
WD	78.6
WD + WD + Max	<u>82.9</u>
NCM	80.1
PC (Ours)	83.7

Table 9: Comparison to state-of-the-art on CIFAR10-LT with imbalance ratio 100. Prototype classifier achieves superior results.