This ICCV paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Distributional Robustness Loss for Long-tail Learning

Dvir Samuel
Bar-Ilan University, Israel
dvirsamuel@gmail.com

Gal Chechik
Bar-Ilan University, Israel
NVIDIA Research, Israel
gal.chechik@biu.ac.il

Abstract

Real-world data is often unbalanced and long-tailed, but deep models struggle to recognize rare classes in the presence of frequent classes. To address unbalanced data, most studies try balancing the data, the loss, or the classifier to reduce classification bias towards head classes. Far less attention has been given to the latent representations learned with unbalanced data. We show that the feature extractor part of deep networks suffers greatly from this bias. We propose a new loss based on robustness theory, which encourages the model to learn high-quality representations for both head and tail classes. While the general form of the robustness loss may be hard to compute, we further derive an easy-to-compute upper bound that can be minimized efficiently. This procedure reduces representation bias towards head classes in the feature space and achieves new SOTA results on CIFAR100-LT, ImageNet-LT, and iNaturalist long-tail benchmarks. We find that training with robustness increases recognition accuracy of tail classes while largely maintaining the accuracy of head classes. The new robustness loss can be combined with various classifier balancing techniques and can be applied to representations at several layers of the deep model.

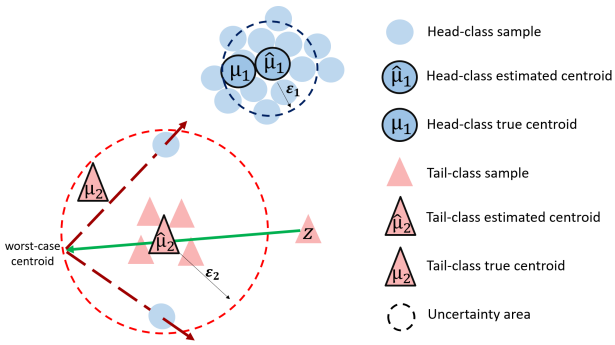

Figure 1: Our *distributional robustness loss* is designed for learning a representation where samples are kept close to the centroid of their class. Here, the empirical centroid $\widehat{\mu_2}$ (framed pink triangle) is estimated based only on few samples (four pink triangles) and as a result it deviates far from the true centroid (μ_2). Our loss pulls the same-class samples (green arrow), and pushes away other-class samples (red arrows). The loss takes into account the estimation error by pushing and pulling towards a worst-case possible distribution within an uncertainty area around the estimated centroid (dashed red line). Uncertainty areas are typically larger for tail classes, compared with head classes that have many samples (blue dashed line around $\widehat{\mu_1}$).

1. Introduction

Real-world data typically has a long-tailed distribution over semantic classes: few classes are highly frequent, while many classes are only rarely encountered. When trained with such unbalanced data, deep models tend to produce predictions that are biased and over-confident towards head classes and fail to recognize tail classes.

Early approaches for handling unbalanced data used re-sampling [12, 17] or loss reweighing [18, 27] aiming to re-balance the training process. Other approaches address unbalanced data by transferring information from head to tail classes [31, 46, 41, 29], or by applying an adaptive loss [6] or regularization of classifiers [21]. The main focus of these techniques is on balancing the multi-class classifier.

Far less attention has been given to the latent representations learned with unbalanced data. Intuitively, head classes are encountered more often during training and are expected to dominate the latent representation at the top layer of a deep model. Counter to this intuition, [21] compared a series of techniques for rebalancing representations and concluded that *"data imbalance might not be an issue in learning high-quality representations"*, and that strong long-tailed recognition can be achieved by only adjusting the classifier. However, the effect of unbalanced data on the learned representations is far from being understood, and the extent to which it may hurt classification accuracy is unknown. While existing rebalancing methods do not improve representa-

tions, the question remains if better representations could substantially improve recognition with unbalanced data.

The current paper focuses on improving the representation layer of deep models trained with unbalanced data. We show that large gains in accuracy can be achieved simply by balancing the representation at the last layer. The key insight is that the distribution of training samples of tail classes does not represent well the true distribution of the data. This yields representations that hinder the classifier applied to that representation.

To address this problem, we introduce ideas from robust optimization theory. We design a loss that is more robust to the uncertainty and variability of tail representations. Standard training follows Empirical Risk Minimization (ERM), which is designed to learn models that perform well on the training distribution. However, ERM assumes that the test distribution is the same as the training distribution, and this assumption often breaks with tail classes. In comparison, Distributionally Robust Optimization (DRO) [28, 14, 4] is designed to be robust against likely shifts of the test distribution. It learns classifiers that can handle the worst-case distribution within a neighborhood of the training distribution. Figure 1 illustrates this idea.

In the general case, computing a loss based on a worst-case distribution may be computationally hard. Here, we show how the worst-case loss can be bounded from above, with a bound that has an intuitive form and can be easily minimized. The resulting bound allows us to minimize the DRO loss, which only affects the representation, and to combine it with a standard classification loss, which tunes a classifier on top of the representation.

The main contributions of this paper are:

1. We formulate learning with unbalanced data as a problem of robust optimization and highlight the role of high variance in hindering tail accuracy.
2. We develop a new loss, **DRO-LT Loss**, based on distributional robustness optimization for learning a balanced feature extractor. Training with DRO-LT yields representations that capture well both head and tail classes.
3. We derive an upper bound of the DRO-LT loss that can be computed and optimized efficiently. We further show how to learn the robustness safety margin for each class, jointly with the task, avoiding additional hyper-parameter tuning.
4. We evaluate our approach on four long-tailed visual recognition benchmarks: CIFAR100-LT, CIFAR10-LT, ImageNet-LT, and iNaturalist. Our proposed method consistently achieve superior performance over previous models.

2. Related Work

2.1. Long-tail recognition

Real-world data usually follows a long-tailed distribution, which causes models to favor head classes and overfit tail classes [5, 35]. Previous efforts to address this effect can be divided into four main approaches: Data-manipulation approaches, Loss-manipulation approaches, Two-stage fine-tuning, and ensemble methods.

Data-manipulation (re-sampling): Data-manipulation approaches aim to balance long-tail data. There are three popular techniques of resampling strategies: (1) Over-sampling minority (tail) classes by simply copying samples [7, 17]. (2) Under-sampling majority (head) classes by removing samples [12, 20]. (3) Generating augmented samples to supplement tail classes [2, 8, 24, 7]. While simple and intuitive, *over-sampling* methods suffer from heavy over-fitting on the tail classes, *under-sampling* methods degrade the generalization of models, and *data augmentation* methods are expensive to develop.

Loss-manipulation (re-weighting): Loss reweighting approaches encourage learning of tail classes by setting costs to be non-uniform across classes. For instance, [18] scaled the loss by inverse class frequency. An alternative strategy down-weighs the loss of well-classified examples, preventing easy negatives from dominating the loss [27] or dynamically rescale the cross-entropy loss based on the difficulty to classify a sample [34]. [6] proposed to encourage larger margins for rare classes. [22] use class-uncertainty information, using Bayesian uncertainty estimates, to learn robust features, and [10] reweighs using the effective number of samples instead of proportional frequency.

Two-stage fine-tuning: Two-stage methods separate the training process into representation learning and classifier learning [32, 21, 30, 35]. The first stage aims to learn good representations from unmodified long-tailed data, training using conventional cross-entropy without re-sampling or re-weighting. The second stage aims to balance the classifier by freezing the backbone and finetuning the last fully connected layers with re-sampling techniques or by learning to unbias the confidence of the classifier. Those methods basically assume that the bias towards head classes is significant only in the classifier (i.e, last fully connected layer), or that tweaking the classifier layer can correct the underlying biases in the representation.

Ensemble-models: Ensemble models focus on generating a balanced model by assembling and grouping models. Typically, classes are separated into groups, where classes that contain similar training instances are grouped together. Then, individual models focused on each group are assembled to form a multi-expert framework. [46] learned one branch for head classes and another for tail classes, then combine the branches using a soft-fusion procedure. [43]

distilled a unified model from multiple teacher classifiers. Each classifier focuses on the classification of a small and relatively balanced group of classes from the data. [40] described a shared architecture for multiple classifiers, a distribution-aware loss, and an expert routing module. The current paper proposes a two-stage approach for training one model with a single classifier. Thus, we compare our method with non-ensemble approaches.

2.2. Distributionally robust optimization

Distributionally Robust Optimization (DRO) [28, 14, 4] considers the case where the test distribution is different from the train distribution. It does that by defining an uncertainty set of test distributions around the training distribution and learning a classifier that minimizes the expected risk with respect to the worst-case test distribution. DRO has been shown to be equivalent to standard robust optimization (RO) [44] and has a strong relationship with regularization methods [36], Risk-Aversion methods and game theory [33]. For more details, see [33]. As far as we know, DRO theory was not applied for long-tail learning.

Many studies looked into learning representations that are robust to adversarial attacks [3, 1, 16] but this is outside the scope of the current paper.

3. Overview of our approach

We start with an overview of the main idea of our approach and provide the details in subsequent sections.

Our goal is to learn a representation at the last layer of a deep network, such that the distributions of samples of different classes are well separated. Then, they can later be correctly classified by different linear classifiers. Deep networks can learn such representations efficiently when trained with enough labeled data. However, when a class has only few samples, the distribution of training samples may not represent well the true distribution of the data, and the representations learned by the model hinder the classifier.

To remedy this shortcoming, we design a loss that is applied to the representation, which takes into account errors due to a small number of samples. Figure 1 illustrates this idea. Our loss extends standard contrastive losses, which pull samples closer to the centroid of their own class and push away samples that belong to other classes. Our new loss accounts for the fact that the true centroids are unknown, and their estimate is noisy. It, therefore, optimizes against the worst possible centroids within a safety hyper ball around the empirical centroid.

In the general case, computing such a worst-case loss may be computationally hard. We further derive an upper bound of that loss that can be computed easily and show that using that bound as a surrogate loss, yields significantly better representations. The resulting loss has a simple general form, yielding that the loss for a sample z is

$$\mathcal{L}(z)_{Robust} = -\log \frac{e^{-d(\widehat{\mu_c},z)-\Delta}}{\sum_{z'} e^{-d(\widehat{\mu_c},z')-\Delta'}}. \tag{1}$$

where $d(\widehat{\mu_c}, z)$ measures the distance in feature space between a sample z and the estimated centroid of its class $\widehat{\mu_c}$, Δ and Δ' are robustness margins that we describe below.

4. Distributional Robust Optimization

When learning a classifier, we seek a model f that minimizes the expected loss for the data distribution $P(x, y)$.

$$\min_f E_{x\sim P}[l_f(x)] = \int l(f(x), y) dP(x, y). \tag{2}$$

Since the data distribution is not known, Empirical Risk Minimization (ERM) [39] proposes to use the training data for an empirical estimate of the data distribution $P_\delta = \frac{1}{n}\sum \delta(x = x_i, y = y_i)$, where δ is the kronecker delta.

$$\textbf{ERM:} \quad \min_f E_{(x,y)\sim P_\delta}\left[l_f(x)\right] \tag{3}$$

Unfortunately, using P_δ to approximate P makes the naive assumption that the test distribution would be close to the empirical train distribution. That assumption may be far from true when the training data is small. In those cases, it is beneficial to choose other estimates of P that are more likely to reduce the loss over the test distribution.

One such solution is given by *Distributionally robust optimization* (DRO) [14, 4]. It suggests learning a model f that minimizes the loss within a family of possible distributions

$$\textbf{DRO:} \quad \min_f \sup_{Q\in U} E_{(x,y)\sim Q}[l_f(x)]. \tag{4}$$

DRO aims to perform well simultaneously for a set of test distributions, each in an uncertainty set U. The set of distributions U is typically chosen as a hyper ball of radius ϵ (ϵ-ball) around the empirical training distribution $\widehat{P}$:

$$U := \{Q : D(Q, \widehat{P}) \leq \epsilon\}, \tag{5}$$

where D is a discrepancy measure between distributions, usually chosen to be the KL-divergence or the Wasserstein distance.

5. Our approach

We now formally describe our approach: **DRO-LT Loss**.

5.1. Preliminaries

We are given n labeled samples (x_i, y_i) $i = 1, \ldots, n$, where y_i is one of k classes $\{c_1, \ldots, c_k\}$. Let f_θ be a feature extractor function f with learnable parameters θ, which maps

any given input sample x_i to $z_i = f_\theta(x_i)$. The set of mapped input $Z = z_1, ...z_n$ resides in some latent vector space.

Let $S_c = \{z_i | y_i = c\}$ be the set of feature vectors representing samples whose label is c. We denote by $\widehat{\mu_c}$ the empirical centroid of that set, namely the mean of all feature vectors z of class c, $\widehat{\mu_c} = \frac{1}{|c|}\sum_{i=1}^{|c|} z_i$. We also denote by μ_c the mean of the true data distribution P of samples from class c, $\mu_c = E_{x\sim P|y=c}[z]$.

5.2. A loss for representation learning

We adopt ideas from metric learning and representation learning [42, 26, 37] for designing a representation-learning loss.

Given a representation z_i of a sample x_i that has a label $y_i = c$, we wish to design a loss that maps z_i close in feature space to other samples from the same class, and far from samples of other classes. We start by developing a contrastive loss and later extend it to a robust one.

Consider a sample (x_i, y_i) from a class $y_i = c$, whose feature representation is z_i. We model the samples of class c as if they are distributed around a centroid μ_c, and the likelihood of a sample decays exponentially with the distance from the centroid μ_c of class c. Such exponential decay has been long used in metric learning and can be viewed as reflecting a random walk over the representation space [15, 13]. The normalized likelihood of a sample is, therefore:

$$P(z_i|\mu_c) = \frac{e^{-d(\mu_c, z_i)}}{\sum_{z'\in Z} e^{-d(\mu_c, z')}}, \tag{6}$$

where d is a measure of divergence or distance between the centroid μ_c and a feature representation z_i. d is typically set to be the Euclidean distance, but can also be modeled with heavier tails as when using a student t-distribution [38].

Similarly, for a set $S_c = \{z_i | y_i = c\}$, the log-likelihood is:

$$\log P(S_c|\mu_c) = \sum_{z_i\in S_c} \log \frac{e^{-d(\mu_c, z_i)}}{\sum_{z'\in Z} e^{-d(\mu_c, z')}}. \tag{7}$$

We define a negative log-likelihood loss as a weighted average over per-class losses:

$$\begin{aligned} \mathcal{L}_{\mathcal{NLL}} &= \sum_{c\in C} w(c)\,(-\log P(S_c|\mu_c)) \qquad (8)\\ &= -\sum_{c\in C} w(c) \sum_{z_i\in S_c} \log \frac{e^{-d(\mu_c, z_i)}}{\sum_{z'\in Z} e^{-d(\mu_c, z')}}, \end{aligned}$$

where $w(c)$ are class weights. Setting $w(c) = \frac{1}{|S_c|}$ gives equal weighting to all classes and prevents head classes from dominating the loss.

5.3. A robust loss

The log-likelihood loss of Eq. (7) operates under the assumption that the centroid of each class μ_c is known. In reality, it is not available to us. Following Empirical-risk minimization, we could naively plug in the empirical estimate $\widehat{\mu_c}$ in Eq. (7), but $\widehat{\mu_c}$ may be a poor approximation of μ_c, and estimation error grows as the number of samples $|S_c|$ decreases. As a result, the log-likelihood and the loss would also be badly estimated.

Instead of approximating the NLL loss ($-\log P(S_c|\mu_c)$), we show that we can bound it with high probability by computing the worst-case loss over a set of candidate centroids. For that purpose, we take an approach based on distributionally robust optimization.

Let $\widehat{p_c}$ be the empirical distribution of samples of class c. We define the uncertainty set of candidate distributions in its neighborhood to be

$$U_c := \{q | D(q||\widehat{p_c}) \le \epsilon_c\}, \tag{9}$$

where D is a measure of divergence of two distributions. Specifically, we consider here the case where D is the Kullback-Leibler divergence, and the distributions under consideration are same-variance spherical Gaussians. In this case, their divergence equals $D_{KL}(q||\widehat{p_c}) = \frac{1}{2\sigma^2} d(\mu_q, \mu_{\widehat{p_c}})^2$[9], where d is the Euclidean distance. Hence for any $q \in U_c$, we have

$$d(\mu_q, \mu_{\widehat{p_c}}) \le \sigma_c\sqrt{2\epsilon_c} \equiv \varepsilon_c, \tag{10}$$

where we define $\varepsilon_c \equiv \sigma_c\sqrt{2\epsilon_c}$ for convenience.

We now derive an upper bound on the NLL loss that we can compute using the estimated centroids $\widehat{\mu_c}$. The form of the bound is very intuitive, it can be viewed as a modification of the NLL loss, where the distance between a sample and an empirical centroid, is increased by a factor that depends on the radius of the uncertainty ball.

Theorem 1. *Let ϵ_c be the radius of the uncertainty set U_c and let σ_c be the variance of the distribution of a sample in class c. Let $p(\epsilon)$ be the probability that the true distribution whose centroid is μ_c is within U_c. The per-class negative log-likelihood is bounded by*

$$-\log P(z|\mu_c) \le -\log \frac{e^{-d(\widehat{\mu_c}, z) - 2\varepsilon_c}}{\sum_{z'\in Z} e^{-d(\widehat{\mu_c}, z') - 2\varepsilon_c \delta(z', c)}} \tag{11}$$

with probability $p(\epsilon)$, where $\delta(z, c) = 1$ if z is of the class c and 0 otherwise and $\varepsilon_c = \sigma_c\sqrt{2\epsilon_c}$.

Proof. Denote the negative log-likelihood of a given sample z and class distribution p with a centroid μ by $\mathcal{L}(z, p) = -\log P(z|\mu)$. With probability $p(\epsilon)$, the true distribution p_c is within the uncertainty ball $p_c \in U_c$. Therefore

$$\mathcal{L}(z, p_c) \le \max_{q\in U_c} \mathcal{L}(z, q) = \max_{q\in U_c}(-\log P(z|\mu_c^q)). \tag{12}$$

Here, μ_c^q is the centroid of a distribution $q \in U_c$. To bound $-\log P(z|\mu_c^q)$, we use the triangle inequality and write:

$$\begin{aligned} d(\mu_c^q, z) &\leq d(\widehat{\mu_c}, z) + d(\widehat{\mu_c}, \mu_c^q) \\ d(\widehat{\mu_c}, z) &\leq d(\mu_c^q, z) + d(\widehat{\mu_c}, \mu_c^q) \end{aligned} \tag{13}$$

yielding

$$-d(\mu_c^q, z) \geq -d(\widehat{\mu_c}, z) - d(\widehat{\mu_c}, \mu_c^q) \geq -d(\widehat{\mu_c}, z) - \varepsilon_c \tag{14}$$

$$-d(\mu_c^q, z) \leq -d(\widehat{\mu_c}, z) + d(\widehat{\mu_c}, \mu_c^q) \leq -d(\widehat{\mu_c}, z) + \varepsilon_c \tag{15}$$

Applying Eq. (14) to the numerator of Eq. (6), and applying Eq. (15) to its denominator, we obtain:

$$P(z|\mu_c^q) = \frac{e^{-d(\mu_c^q, z)}}{\sum_{z' \in Z} e^{-d(\mu_c^q, z')}} \geq \frac{e^{-d(\widehat{\mu_c}, z) - 2\varepsilon_c}}{\sum_{z' \in Z} e^{-d(\widehat{\mu_c}, z') - 2\varepsilon_c \delta(z', c)}} \tag{16}$$

where $\delta(z, c) = 1$ when c is the class of z and 0 otherwise. This inequality holds for all distributions $q \in U_c$, and also for the true distribution μ_c as a special case (with probability $p(\epsilon)$). The negative log-likelihood is therefore bounded by

$$-\log P(z|\mu_c) \leq -\log \frac{e^{-d(\widehat{\mu_c}, z) - 2\varepsilon_c}}{\sum_{z' \in Z} e^{-d(\widehat{\mu_c}, z') - 2\varepsilon_c \delta(z', c)}}. \tag{17}$$

This completes the proof. □

Based on the theorem, define a surrogate robustness loss:

$$\mathcal{L}_{Robust} = -\sum_{c \in C} w(c) \sum_{z \in S_c} \log \frac{e^{-d(\widehat{\mu_c}, z) - 2\varepsilon_c}}{\sum_{z'} e^{-d(\widehat{\mu_c}, z') - 2\varepsilon_c \delta(z', c)}} \tag{18}$$

This surrogate loss amends the loss of Eq. (8) in a simple and intuitive way. It increases the distance $d(\widehat{\mu_c}, z)$ between a sample and an empirical centroid in a way that depends on the radius of the uncertainty ball of a class.

Joint loss. In practice, we train the deep network with a combination of two losses. A standard cross-entropy loss is applied to the output of the classification layer, and the robust loss is applied to the latent representation of the penultimate layer. We linearly combine these two losses

$$\mathcal{L} = \lambda \mathcal{L}_{CE} + (1 - \lambda) \mathcal{L}_{Robust}. \tag{19}$$

The trade-off parameter λ can be tuned using a validation set. See implementation details below.

$p(\varepsilon)$ **and Lower bound.** Appendix A provides a formal definition of $p(\varepsilon)$. Appendix B derives a lower bound of our loss and also shows empirically that the bounds are tight in our experiments.

6. Training

Uncertainty radius: The size of the uncertainty ϵ-ball around each class plays an important role. When the uncertainty radius is too small, the probability that the true centroid is within the uncertainty area decreases, together with the probability that the bound holds. When the radius is too large, the bound is more likely to hold, but it becomes less tight. Furthermore, since tail classes have fewer samples, the estimate of class centroids is expected to be noisier and a larger radius is needed.

We explored three ways to determine the radius:

1. **Shared ε:** All classes share the same uncertainty radius.
2. **Sample count $\varepsilon/\sqrt{n}$:** The class radius scales with $1/\sqrt{n}$, where n is the number of training samples. This scaling is based on the fact that the standard-error-of-the-mean decays as $\sqrt{n}$, and leads to tail classes having a larger safety radius. See Appendix C for more details.
3. **Learned ε:** We treated the radius as a learnable parameter and tuned its value during training.

In the first two cases, the radius parameter ε was treated as a hyper-parameter and tuned using a validation set.

Training process: To compute the robustness loss, an initial feature representation of the data is required for estimating class centroids. As a result, during training, we first train the model with standard cross-entropy loss ($\lambda = 1$) to learn initial feature representations and centroids. Then, we add the DRO loss to the training by setting $\lambda < 1$. Finally, as in [21], we learn a balanced classifier by freezing the feature extractor and fine-tune only the classifier with balanced sampling.

Estimating centroids: Calculating the centroids for each class is computationally expensive if computed often using the full dataset. One could estimate the centroids within each batch, but with unbalanced data, minority classes would hardly have any samples and their centroid estimates would be very poor. Instead, we compute the features z_i for every sample x_i at the beginning of every epoch, compute the centroids for each class, and keep them fixed in memory for the duration of the epoch.

7. Experiments

7.1. Datasets

We evaluated our proposed method using experiments on three major long-tailed recognition benchmarks.

(1) CIFAR100-LT [6]: A long-tailed version of CIFAR100 [25]. CIFAR100 consists of 60K images from 100 categories. Following [6], we control the degree of data imbalance with an imbalance factor β. $\beta = \frac{N_{max}}{N_{min}}$, where N_{max} and N_{min} are the number of training samples for the most frequent and the least frequent classes, respectively.

Imbalance Type	Long-tailed CIFAR-100		
Imbalance Ratio	100	50	10
CE*	38.32	43.85	55.71
ReSample [10]	33.44	-	55.06
ReWeight [10]	33.99	-	57.12
Focal Loss [27]	38.41	44.32	55.78
LDAM Loss [6]	39.60	44.97	56.91
τ-norm* [21]	41.11	46.74	57.06
cRT* [21]	41.24	46.83	57.93
smDRAGON [35]	43.55	46.85	58.01
SSL* [23]	37.51	44.02	56.70
SSP [45]	43.43	47.11	58.91
DRO-LT (Ours)			
$\varepsilon = 0$ (ERM)	43.92	52.31	59.54
Shared ε	45.66	55.32	61.22
$\varepsilon/\sqrt{n}$	46.92	57.20	63.10
Learned ε	**47.31**	**57.57**	**63.41**

Table 1: Top-1 accuracy of ResNet32 on long-tailed CIFAR-100 [6], comparing our method and SoTA techniques. Asterisks * denote reproduced results. DRO-LT variants achieve best results on all imbalance ratios.

We conduct experiments with $\beta = 100$, 50, and 10.
(2) ImageNet-LT [31]: A long-tailed version of the large-scale object classification dataset ImageNet [11] by sampling a subset following the Pareto distribution with power value $\alpha = 6$. Consists of 115.8K images from 1000 categories with 1280 to 5 images per class. **(3) iNaturalist** [19]: A large-scale dataset for species classification. It is long-tailed by nature, with an extremely unbalanced label distribution. Has 437.5K images from 8,142 categories.

7.2. Compared methods

We compared our approach with the following methods. ***(A) Baselines:*** **CE:** Naive training with a cross-entropy loss; **ReSample:** Over-sampling classes to reach a uniform distribution as in [6]. ***(B) Loss-manipulation:*** **Reweight:** Reweight the loss as in [6, 46]; **Focal Loss** [27] and **LDAM Loss** [6]. ***(C) Two-stage fine-tuning:*** **τ-norm** [21], **cRT** [21] and **smDragon** [35]. ***(D) Representation learning:*** SSL the method by [23] and SSP, the method by [45].
(E) DRO-LT variants: We compared four ways to set the radii of U_c. Set $\varepsilon = 0$ (ERM); ε value shared across all classes; A shared value divided by the square root of class size ($\varepsilon/\sqrt{n}$); and a learned value per class.

7.3. Evaluation Protocol

Following [31] and [21], we report the top-1 accuracy over all classes on class balanced test sets. This metric is denoted by *"Acc"*. For CIFAR-100 and ImageNet-LT, we further report accuracy on three splits of the set of classes. *"Many":* classes with more than 100 train samples; *"Med":* classes with 20 – 100 train samples; and *"Few":* classes with less than 20 train samples.

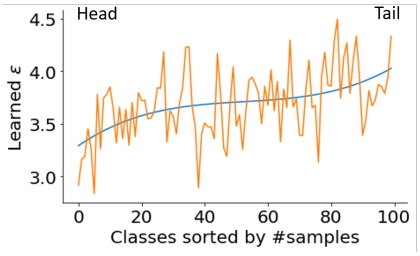

Figure 2: Learned ε values for each class of a ResNet-32 trained on CIFAR100-LT with imbalance ratio 50. Blue: Polynomial fit.

7.4. Implementation details

In all experiments, we use an SGD optimizer with a momentum of 0.9 to optimize the network. For long-tailed CIFAR-100, we follow [6] and train a ResNet-32 backbone on one GPU with a multistep learning-rate schedule. For ImageNet-LT and iNaturalist, we follow [21] and use the cosine learning-rate schedule to train a ResNet-50 backbone on 4 GPUs.

Hyper-parameter tuning: We determined the number of training epochs (early-stopping), and tuned hyper-parameters using the validation set. We optimized the following hyper-parameters: (1) Radius parameter $\varepsilon \in \{1, 2, 5, 10, 30, 70\}$ for "Shared ε" and "Sample Count $\varepsilon/\sqrt{n}$". (2) Trade-off parameter $\lambda \in \{0, 0.3, 0.5, 0.7, 1\}$. (3) Learning rate $\in \{10^{-4}, 10^{-3}, 10^{-2}\}$. we studied the sensitivity of the accuracy to the values of ε and λ, and found that high accuracy is obtained for a wide range of ϵ values. We also found accuracy to be quite stable when tuning λ and selected $\lambda = 0.5$. See Appendix D for details.

8. Results

CIFAR100-LT: Table 1 compares DRO-LT with common long-tail methods on CIFAR100-LT. It shows that all our robust loss variants consistently achieve the best results on all imbalance factors, emphasizing the importance of robust learning in unbalanced data. "Learned ε" outperforms other variants of our method. **CIFAR-10-LT:** Appendix G provides results for CIFAR-10-LT with imbalanced ratio 100. **ImageNet-LT:** Table 2 further evaluates our approach on ImageNet-LT and CIFAR100-LT (imbalance factor=100) reporting accuracy for different test splits. DRO-LT performs well on tail classes ("Few") as well as head classes ("Many"). This contrasts with previous methods that sacrifice head accuracy for better tail classification. This also suggests that DRO-LT learns high-quality features for all classes. **iNaturalist:** Table 3 evaluates our approach on the large-scale iNaturalist. DRO-LT slightly improves the accuracy compared with SoTA baselines. See Appendix F for more results and further analysis.

	Long-tailed CIFAR-100				Long-tailed ImageNet			
Methods	Many	Med	Few	Acc	Many	Med	Few	Acc
CE*	**65.5**	37.9	7.4	38.3	**64.0**	38.8	5.8	41.6
LDAM Loss [6]	61.0	41.6	19.8	39.6	-	-	-	-
OLTR [30]	61.8	41.4	17.6	41.2	-	-	-	-
τ-norm [21]	61.4	42.5	15.7	41.1	56.6	44.2	27.4	46.7
smDragon [35]	60.5	44.3	23.5	43.5	59.7	44.2	25.3	47.4
SSL * [23]	64.1	36.9	7.1	37.5	61.4	47.0	28.2	49.8
DRO-LT (Ours)								
$\varepsilon = 0$ (ERM)	61.9	43.7	22.2	43.9	61.0	45.5	26.8	48.1
Shared ε	64.1	47.9	21.5	45.7 ± 0.2	62.6	45.2	30.5	51.6 ± 0.4
$\varepsilon/\sqrt{n}$	65.0	49.8	22.3	46.9 ± 0.2	63.8	49.5	32.7	53.0 ± 0.3
Learnable ε	64.7	**50.0**	**23.8**	$\mathbf{47.3 \pm 0.1}$	**64.0**	**49.8**	**33.1**	$\mathbf{53.5 \pm 0.2}$

Table 2: Top-1 accuracy on long-tailed CIFAR-100 [6] with imbalance factor 100, and ImageNet-LT[31]. We also report many-shot ("Many"), medium-shot ("Med") and few-show ("Few") performance separately. Our method performs well on tail classes without sacrificing head accuracy. Asterisks * denote results reproduced using code published by authors.

	iNaturalist
CE	61.7
LDAM Loss [6]	68.0
$\tau-$norm [21]	69.5
CB LWS [21]	69.5
smDragon [35]	69.1
SSL * [23]	66.4
DRO-LT (ours)	
Learned ε	$\mathbf{69.7 \pm 0.1}$

Table 3: Top-1 accuracy on long-tailed iNaturalist. DRO-LT achieves slightly better results compared to previous methods. Asterisks * denote reproduced results by us.

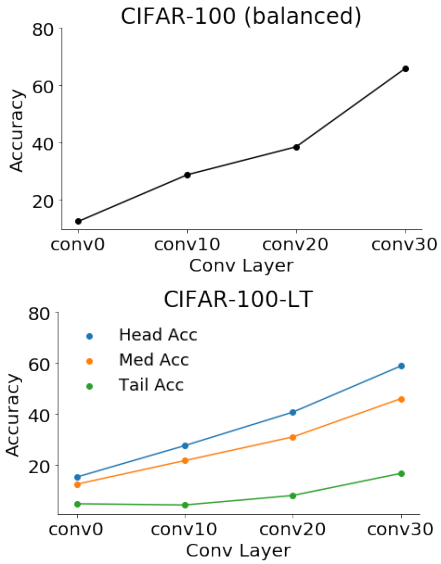

Figure 3: Accuracy of a nearest-centroid neighbor classifier when applied to convolutional layers 0, 10, 20, and 30 of a ResNet-32. **Top**: The model was trained on balanced CIFAR-100. The validation accuracy grows when using higher layers. **Bottom:** The model was trained on CIFAR-100-LT. We report balanced validation accuracy for head classes (blue), medium classes (orange) and tail classes (green). The accuracy gap between head and tail classes is substantial even at all layers.

Classifier vs feature extractor: Our method focuses on improving the learned representation at the penultimate layer. Other methods focused on improving the classifier applied to that representation. Therefore, it is interesting to explore the relation between these two tasks (compare with [46, 21]). We, therefore, compared different feature extractors and classifier training methods.

For representation learning, we employ plain training with a cross-entropy loss (**CE**), balanced resampling of the data (**RS**) and our method (**DRO-LT**). For classifier learning, we freeze the parameters of the feature extractor and fine-tune the classifier in three ways: cross-entropy loss (**CE**), re-sampling (**RS**) following the protocol of [6], and balanced classifier (**LWS**) [21]. Table 4 provides the top-1 accuracy of all combinations. It shows that our representation learning approach enables all types of classifiers to reach good performance, compared to other representation learning approaches. This strongly suggests that our method learns: (1) good feature representations for both head and tail classes, and (2) to separate them in a way that discriminative classifiers can easily distinguish between classes.

Adaptive robustness: Figure 2 shows the values of uncertainty per-class radii (ε) that were learned by training a ResNet-32 on CIFAR100-LT with an imbalance ratio of 50. The model learned slightly larger radii for tail classes compared with head classes. See the supplemental for details about the effect of the radius on accuracy.

Robustness: Our loss is expected to improve recognition mostly at tail classes. Figure 4 compares train error and test error between a model trained with cross-entropy loss (red) and a model trained with our approach (blue). Using a robustness loss cuts down errors substantially, in tail classes, without hurting head classes.

Feature space visualization: To gain additional insight, we look at the t-SNE projection of learned representations and compared vanilla cross-entropy loss with our proposed method. Figure 5 shows that our learned feature space is

Representation learning	Classifier learning		
	CE	RS (cRT)	LWS [21]
CE	38.5	41.2	41.4
RS	34.9	37.6	36.3
DRO-LT (**ours**)	**41.2**	**47.3**	**46.8**

Table 4: Top-1 accuracy of different representation learning manners and classifier learning manners, on CIFAR100-LT. CE refers to Cross-Entropy, RS refers resampling and LWS refers to balanced classifier according to [21]. The results suggest that simple resampling methods achieve good results when the learned features are good for both head and tail.

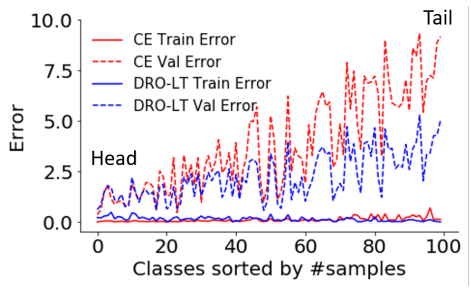

Figure 4: Comparing train and test error between a model trained with vanilla cross-entropy loss and a model trained with our DRO-LT. The gap between the train error and test error for tail classes is much lower in DRO-LT compared with vanilla, while remaining the same for head classes.

more compact with margins around head and tail classes. Tail classes have larger margins since the estimation of their features is less accurate.

How unbalanced are latent representations? The above analysis focused on correcting the representation at the penultimate layer. But how biased are the representations at lower layers? Intuitively, the low layers represent more "physical" properties of inputs, while higher layers capture properties that correspond more to semantic classes. One would expect that early layers would be quite balanced.

We tested class imbalance in several layers of a ResNet-32 in the following way. We first trained a ResNet-32 on the unbalanced CIFAR100-LT dataset. Then, for each latent representation, we computed the centroids of each class and used them to classify all samples using a nearest-centroid classifier. Figure 3 shows the accuracy obtained with this classifier when applied to layers 0, 10, 20, and 30 of the ResNet-32. When training with balanced data (left), the accuracy grows when using higher layers, as expected. Surprisingly, when training with unbalanced data (right), there is a substantial difference in accuracy achieved for head and tail classes in every layer that we tested. While accuracy grows when using higher layers, accuracy differences between head and tail classes are maintained, even in low layers that are believed to represent class agnostic features. In Appendix H,

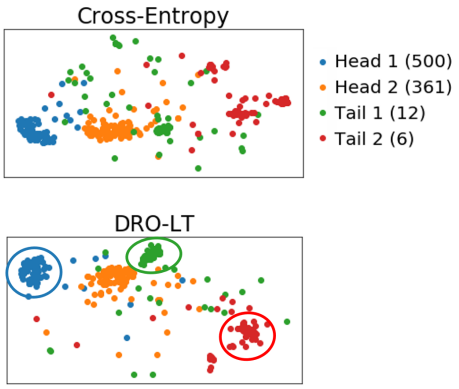

Figure 5: t-SNE visualization of embedding space of CIFAR100-LT obtained using cross-entropy loss and DRO-LT Loss method. The feature embedding of our model is more compact for both head (blue) and tail (green and red) classes and better separated. The number of samples for each class is written in parentheses.

we compare the accuracy of a nearest-centroid neighbor classifier between a model trained with standard cross-entropy loss and one trained with DRO-LT. We show that DRO-LT narrows the accuracy gap between head and tail.

9. Discussion

This paper investigates the feature representations learned by deep models trained on long-tail data. We find that such models suffer greatly from bias towards head classes in their feature extractor (backbone), which hurts recognition. This is in contrast to previous studies suggesting that unbalanced data does not hurt representation learning and re-balancing the classifier layer is sufficient. To learn a balanced representation, we take a robustness approach and develop a novel loss based on Distributionally Robust Optimization (DRO) theory. We further derive an upper bound of that loss that can be minimized efficiently. We show how the robustness safety margin can be learned during training, and do not require additional hyper-parameter tuning.

Training with a combination of the DRO-LT loss and a standard classifier sets new state-of-the-art results on three long-tail benchmarks: CIFAR100-LT, ImageNet-LT, and iNaturalist. Our method not only improves the performance of tail classes but also maintains high accuracy at the head. These results suggest that proper training of representations for unbalanced data can have a large impact on downstream accuracy. We believe that our finding not only contributes to a deeper understanding of the long-tailed recognition task but can offer inspiration for future work.

Acknowledgments: This work was funded by the Israeli innovation authority through the AVATAR consortium; by the Israel Science Foundation (ISF grant 737/2018); and by an equipment grant to GC and Bar Ilan University (ISF grant 2332/18).

References

[1] Anurag Arnab, Ondrej Miksik, and Philip HS Torr. On the robustness of semantic segmentation models to adversarial attacks. In *CVPR*, 2018.

[2] S. Beery, Y. Liu, D. Morris, J. Piavis, A. Kapoor, M. Meister, and P. Perona. Synthetic examples improve generalization for rare classes. *Preprint arXiv:1904.05916*, 2019.

[3] Dimitris Bertsimas, Jack Dunn, Colin Pawlowski, and Ying Daisy Zhuo. Robust classification. *INFORMS Journal on Optimization*, 2019.

[4] Dimitris Bertsimas, Vishal Gupta, and Nathan Kallus. Data-driven robust optimization. *Mathematical Programming*, 2018.

[5] M. Buda, A. Maki, and M. Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 2018.

[6] K. Cao, C. Wei, A. Gaidon, N. Arechiga, and T. Ma. Learning imbalanced datasets with label-distribution-aware margin loss. *NeurIPS*, 2019.

[7] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 2002.

[8] Peng Chu, Xiao Bian, Shaopeng Liu, and Haibin Ling. Feature space augmentation for long-tailed data. *arXiv preprint arXiv:2008.03673*, 2020.

[9] T. Cover and J. Thomas. Elements of information theory. 1991.

[10] Y. Cui, M. Jia, T. Lin, Y. Song, and S. Belongie. Class-balanced loss based on effective number of samples. *CVPR*, 2019.

[11] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li. Imagenet: A large-scale hierarchical image database. *CVPR*, 2009.

[12] Chris Drummond. C 4 . 5 , class imbalance , and cost sensitivity : Why under-sampling beats oversampling. 2003.

[13] Amir Globerson, Gal Chechik, Fernando CN Pereira, and Naftali Tishby. Euclidean embedding of co-occurrence data. In *NIPS*, 2004.

[14] Joel Goh and Melvyn Sim. Distributionally robust optimization and its tractable approximations. *Operations research*, 2010.

[15] Jacob Goldberger, Geoffrey E Hinton, Sam Roweis, and Russ R Salakhutdinov. Neighbourhood components analysis. *Advances in neural information processing systems*, 2004.

[16] Gaurav Goswami, Nalini Ratha, Akshay Agarwal, Richa Singh, and Mayank Vatsa. Unravelling robustness of deep learning based face recognition against adversarial attacks. In *AAAI*, 2018.

[17] H. Han, W. Wang, and B. Mao. Borderline-smote: A new over-sampling method in imbalanced data sets learning. In *ICIC*, 2005.

[18] H. He and E. A. Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 2009.

[19] G. Horn, O. Aodha, Y. Song, A. Shepard, H. Adam, P. Perona, and S. Belongie. The inaturalist challenge 2017 dataset. *ArXiv*, 2017.

[20] Xinting Hu, Yi Jiang, Kaihua Tang, Jingyuan Chen, Chunyan Miao, and Hanwang Zhang. Learning to segment the tail. In *CVPR*, 2020.

[21] B. Kang, S. Xie, M. Rohrbach, M. Yan, A. Gordo, J. Feng, and Y. Kalantidis. Decoupling representation and classifier for long-tailed recognition. *ICLR*, 2020.

[22] Salman Khan, Munawar Hayat, Syed Waqas Zamir, Jianbing Shen, and Ling Shao. Striking the right balance with uncertainty. In *CVPR*, 2019.

[23] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, A. Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Nuerips*, abs/2004.11362, 2020.

[24] Jaehyung Kim, Jongheon Jeong, and Jinwoo Shin. M2m: Imbalanced classification via major-to-minor translation. *CVPR*, 2020.

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

[26] Brian Kulis et al. Metric learning: A survey. *Foundations and trends in machine learning*, 2012.

[27] T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. *ICCV*, 2017.

[28] Anqi Liu and Brian D. Ziebart. Robust classification under sample selection bias. In *NIPS*, 2014.

[29] Jialun Liu, Yifan Sun, Chuchu Han, Zhaopeng Dou, and Wenhui Li. Deep representation learning on long-tailed data: A learnable embedding augmentation perspective. In *CVPR*, 2020.

[30] Z. Liu, Z. Miao, X. Zhan, J. Wang, B. Gong, and S. Yu. Large-scale long-tailed recognition in an open world. In *CVPR*, 2019.

[31] 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.

[32] Wanli Ouyang, X. Wang, Cong Zhang, and X. Yang. Factors in finetuning deep model for object detection with long-tail distribution. *CVPR*, 2016.

[33] H. Rahimian and S. Mehrotra. Distributionally robust optimization: A review. *ArXiv*, 2019.

[34] S. Ryou, S. Jeong, and P. Perona. Anchor loss: Modulating loss scale based on prediction difficulty. In *ICCV*, 2019.

[35] Dvir Samuel, Yuval Atzmon, and Gal Chechik. From generalized zero-shot learning to long-tail with class descriptors. In *WACV*, 2021.

[36] Soroosh Shafieezadeh-Abadeh, Peyman Mohajerin Esfahani, and D. Kuhn. Distributionally robust logistic regression. In *NIPS*, 2015.

[37] Jake Snell, Kevin Swersky, and Richard S Zemel. Prototypical networks for few-shot learning. *NIPS*, 2017.

[38] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using (t-sne). *Journal of Machine Learning Research*, 2008.

[39] Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 2013.

[40] Xudong Wang, Long Lian, Zhongqi Miao, Ziwei Liu, and Stella X Yu. Long-tailed recognition by routing diverse distribution-aware experts. *arXiv preprint arXiv:2010.01809*, 2020.

[41] Yiru Wang, Weihao Gan, Jie Yang, Wei Wu, and Junjie Yan. Dynamic curriculum learning for imbalanced data classification. In *ICCV*, 2019.

[42] Kilian Q Weinberger, John Blitzer, and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. In *Advances in neural information processing systems*, 2006.

[43] Liuyu Xiang and G. Ding. Learning from multiple experts: Self-paced knowledge distillation for long-tailed classification. In *ECCV*, 2020.

[44] Huan Xu, Constantine Caramanis, and Shie Mannor. A distributional interpretation of robust optimization. *Mathematics of Operations Research*, 2012.

[45] Yuzhe Yang and Zhi Xu. Rethinking the value of labels for improving class-imbalanced learning. *NeurIPS*, 2020.

[46] Boyan Zhou, Quan Cui, Xiu-Shen Wei, and Zhao-Min Chen. Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In *CVPR*, 2020.