Learning Debiased Representations for Long-Tailed Visual Recognition

A Major Qualifying Project (MQP) Report
Submitted to the Faculty of
WORCESTER POLYTECHNIC INSTITUTE
in partial fulfillment of the requirements
for the Degree of Bachelor of Science in

Electrical and Computer Engineering,
Computer Science

By:

Prudence Lam

Project Advisors:

Ziming Zhang

Date: April 2023

Abstract

Real-world data often exhibit a long-tailed distribution, posing a challenge for classification models that are inherently biased towards higher frequency classes. Despite extensive research on learning unbiased classifiers, the issue of representation bias remains under-explored. Our observations show a negative correlation between class frequency and intra-class variance in feature space. We explore the use of data augmentation, specifically mixup and implicit semantic data augmentation (ISDA) in learning more uniformly distributed features. Moreover, we use the class-conditional statistics obtained from ISDA to fit linear discriminant analysis (LDA) directly on the features, which we hypothesize to be more robust than Softmax. Extensive experiments demonstrate the competitiveness of our framework on four long-tail benchmarks. Our code can be found at https://github.com/p-lam/LTR-MQP

Contents

1 Introduction **1**

1.1 Contributions . . . 2

2 Related Works **3**

2.1 Data Resampling . . . 3

2.2 Loss Reweighting . . . 3

2.3 Data Augmentation . . . 4

2.4 Two-stage methods . . . 5

3 Methodology **5**

3.1 Problem Definition . . . 5

3.2 Analysis of mixup . . . 6

3.3 Implicit Semantic Data Augmentation (ISDA) . . . 10

3.3.1 Online Covariance Estimation . . . 11

3.4 Classification with Linear Discriminant Analysis (LDA) . . . 12

3.4.1 Theoretical Motivation . . . 12

3.4.2 Linear Discriminant Analaysis . . . 13

3.4.3 Classification Rule . . . 14

4 Experiments **15**

4.1 Experiment Setup . . . 15

4.1.1 Datasets . . . 15

4.1.2 Implementation Details . . . 15

4.2 Ablation Study . . . 16

4.3 Benchmark Results . . . 18

5 Conclusion **19**

5.1 Future Work . . . 20

Appendices **26**

A Visualization of Proposed Framework **26**

List of Tables

1 Ablation study on CIFAR10/100-LT of top-1 accuracy of different methods. . . . 16

2 Top-1 Accuracy on Benchmark Datasets . . . 19

3 Top-1 Accuracy on Benchmark Datasets . . . 19

List of Figures

1 In a long-tailed distribution, few classes have many instances (head), while the majority of the classes have a few (tail). In representation space, head class features are tightly compact, whereas tail class features are diffused. The data here is drawn from CIFAR10-LT. 2

2 2-dim TSNE projection of a tail class along with 2 head classes for CIFAR100-LT. **(a)** For models trained with CE, the head class is relatively compact, while the tail class is diffused. However, the tail class still occupies less space in feature space compared to the head class. **(b)** When mixup is added into training, the tail class is emphasized. **(c)** Our method constrains the variances of all classes to be close to each other. 3

3 When training a model with just CE (see **(a)**, **(c)**, **(e)**), the weights of the head classes grow at a rate much faster than that of the tail classes. This is somewhat ameliorated with mixup, as shown in **(b)**, **(d)**, and **(f)**. The "balancing" effect of mixup is more prominant as the imbalance factor decreases. 8

4 **(a)** Per-class weight norms for a model trained without mixup (*blue*) and with mixup (*orange*) on CIFAR100-LT. **(b)** The intra-class variances for each class given a model trained without mixup (*blue*) and with mixup (*orange*). **(c)** The L2 norms of the difference between per-class covariance matrices without mixup training and **(d)** with mixup training on CIFAR10-LT. The figures are best viewed in color. 9

5 Comparison of the intra-class variances of a model trained with our method (*green*) versus with mixup (*(orange)*) and without (*blue*) on CIFAR100-LT. The left y-axis represents the scale, while the right y-axis denotes the number of samples in a class. 17

6 Comparison of the classifier weight norms of a model trained with our method (*green*) versus with mixup (*(orange)*) and without (*blue*) on CIFAR100-LT. 18

7 Illustration of our method, decoupled into **(a)** Representation Learning and **(b)** Classifier Learning 26

1 Introduction

The field of deep learning has experienced rapid advancements over the past decade, particularly in computer vision, natural language processing, and reinforcement learning. While many of these improvements have come as a result of new model architectures, such as ResNets and ViTs (21; 16) for vision, the success of these models is founded upon newfound access to large labelled datasets, such as ImageNet (14). However, these datasets are often artificially balanced to achieve an approximately even class distribution for model training. This approach is inconsistent with real-world data, which often exhibits a "long-tail" distribution, characterized by a few dominant categories (the "head") and a large number of relatively rare categories (the "tail") (50) . Unfortunately, using standard training techniques on long-tail data is not relibale, as this often results in poor performace on tail categories. As such, long-tailed recognition (LTR) has been introduced as a necessary component for real-world deployment of deep learning models (28; 32; 57).

The main goal of long-tail recognition is to achieve high accuracy across all classes, regardless of how many samples were present in the training set. The ability to detect tail classes is especially relevant for safety-driven applications. For example, many skin cancer classification datasets heavily underrepresent people with darker skin tones (48). Naive training of a classification model on such a dataset may result in poor treatment of patients falling into the tail demographic. In this scenario, and many others, it may be prohibitively expensive or difficult to collect enough tail samples to make the dataset balanced (42). Thus it is imperative to be able to train robust models in spite of long-tail distributions.

In this work, we focus on image classification for long-tailed data, known as long-tailed visual recognition. One common perspective is to mitigate the model bias introduced by the typical training of a softmax classifier, which tends to associate a higher weight norm with head classes. Some recent techniques include data resampling, loss re-weighting, and decoupled training. Data resampling and loss re-weighting methods both look to equalize the impact of each class on the softmax classifier, typically through bootstrapping tail classes or increasing the misclassification cost of tail classes. Inspired by representation learning, recent studies have shown that training policies can be further improved by de-coupling the training of features and the classifier (23). However, these approaches often require careful hyperparameter tuning and may prioritize the learning of the classifier over the features.

We propose a new method that aims to learn an optimal feature representation for long-tailed data. We first demonstrate that there exists a negative correlation between class frequency and intra-class variance. Specifically, we find that samples in head classes tend to be more tightly clustered, while tail classes are more

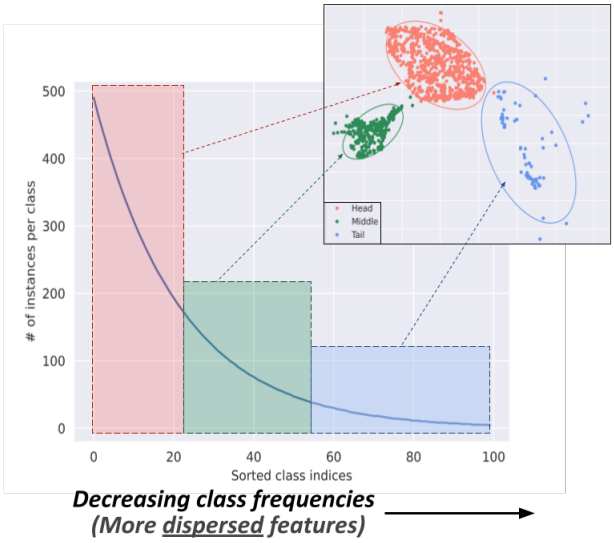

Figure 1: In a long-tailed distribution, few classes have many instances (head), while the majority of the classes have a few (tail). In representation space, head class features are tightly compact, whereas tail class features are diffused. The data here is drawn from CIFAR10-LT.

dispersed, resulting in more challenging classification. Figure 1 illustrates this phenomenon through a t-SNE projection (40) of features produced from a pre-trained model of a head, middle, and tail class. Furthermore, we show in Figure 7b that the use of stronger data augmentations, such as those proposed in (53), (43), and (11), helps ameliorate this issue.

Finally, propose the use of Fisher's Discriminant Analysis (FDA) (19; 47) for classification. This method learns a linear projection to a subspace that can more effectively collapse classes, thereby reducing the impact of noisy features. Our final composition of selected data augmentations and FDA achieves competitive performance with state-of-the-art techniques.

1.1 Contributions

Our core contributions are provided as follows: We complete an extensive literature review with key methods summarized in section 2. We conduct a theoretical analysis of common data-augmentation techniques and classifier objectives under long-tail distributions, leading us to our proposed data augmentation+LDA algorithm described in section 3. We demonstrate the efficacy of our method through extensive empirical benchmarks in section 4. We conclude with a discussion of the project and potential plans for future work in section 5.

(a) Without mixup (b) With mixup (c) Our method

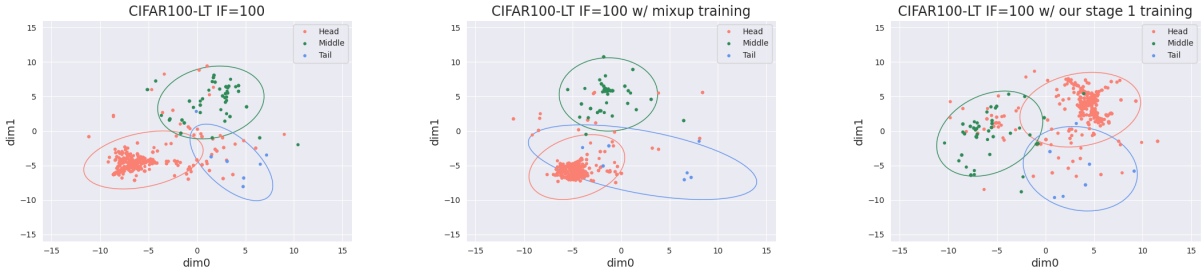

Figure 2: 2-dim TSNE projection of a tail class along with 2 head classes for CIFAR100-LT. **(a)** For models trained with CE, the head class is relatively compact, while the tail class is diffused. However, the tail class still occupies less space in feature space compared to the head class. **(b)** When mixup is added into training, the tail class is emphasized. **(c)** Our method constrains the variances of all classes to be close to each other.

2 Related Works

2.1 Data Resampling

An intuitive way of handling imbalances in long-tailed data is to over-sample the tail classes (5; 33; 6). One such method is *class-aware sampling* (CAS) (33), which ensures an equal proportion of classes within each mini-batch. Similarly, one can under-sample the head classes (5; 22) by removing their instances from training. However, with over-sampling, a model risks over-fitting to tail classes. In contrast, the removal of data in under-sampling can impact a model's generalization ability and lead to under-fitting on head classes. Improved resampling techniques, such as synthetic (9) or meta-sampling (30), can mitigate some of these issues but are computationally expensive, particularly on large datasets. More recently, studies have approached re-sampling from a meta-learning perspective (52). Balanced Meta-Softmax (30), for example, estimates the optimal class sampling rate on a balanced meta validation set.

2.2 Loss Reweighting

Another approach to LTR is to re-weigh classes or instances through the network's objective function (50). Early works sought to reduce bias by adjusting class weights, the most simple being to weigh them proportional to their inverse label frequencies (3). However, this method was found to produce sub-optimal results. To improve upon this, Cui *et al.* (13) proposed the concept of *effective number of samples*, and enforced a class-balanced re-weighting term that is inversely proportional to this value. Balanced Meta-Softmax (30) utilizes the label priors for re-weighting by multiplying prediction logits by their label frequencies.

Tan *et al.* (36; 35) approached the problem from a gradient perspective with the *equalization losses*, which dynamically balance the gradients of the head and tail classes during training.

Instead of using label frequencies, Lin *et al.* (26)considers the difficulty of predicting each class as a basis for re-weighting through Focal Loss. Their intuition is that since tail classes are harder to predict than head classes, their prediction probabilities can be used determine appropriate class weights. Other studies have utilized meta-learning to learn optimal class weights (34; 31). More recently, logit adjustment methods, such as (29; 49) have achieved much success as well, and can be enforced within the CE loss or applied post-hoc.

More recently, supervised contrastive learning (SCL) methods have proved to be extremely effective on long-tailed data. Balanced Contrastive Learning (59) improves on vanilla SCL by equilibrating the gradients of negative classes, and encouraging all classes to appear every mini-batch. PaCo (12) introduces parametric prototypes to reduce model bias on head classes in SCL. TSC (25) enforces a uniform feature distribution by manually calculating a set of "targets" classes should converge to during training. The efficacy of these works serves as a motivation to learn robust representations for long-tailed data.

2.3 Data Augmentation

Data augmentation is frequently employed in deep network training to augment the size and diversity of the training data, and has been previously utilized to improve the representation quality of tail samples. Mixup (53) is a popular regularization technique that trains the model on convex combinations of of input-target pairs. Several studies have observed that this regularization is effective for model calibration (38; 56; 58) and robustness (54). More recent mixup-based approaches include Manifold-Mixup (43), which expands mixup to intermediate feature representations, and Remix (10), which relaxes the mixing of labels to assign more weight to infrequent classes. Unimix (49) further improves upon mixup in long-tailed scenarios by incorporating a dynamic, tail-favored mixing factor and a sampler that encourages the formulation of more h head-tail pairs.

Other augmentation-based works include Liu *et al.* (27), which promotes similar feature "clouds" by augmenting tail classes with the angular variances learned from head classes, and MetaSAug (24) which adapts upon ISDA (46) to the imbalanced case by incorporating meta-learning to estimate class-specific covariances. Despite the remarkable successes of these works, data augmentation remains relatively underexplored in long-tailed recognition. Our proposed method draws inspiration from the aforementioned studies.

2.4 Two-stage methods

Following Kang *et al.* (23), long-tail recognition works separate the training process into feature-learning and classifier-learning stages. The first stage is trained with standard instance-based sampling to learn more general representations, while the second stage employs re-weighting or re-sampling methods to fine-tune the classifier. Recent techniques to improve fine-tuning of the classifier include weight-balancing losses (2), label-aware smoothing (58), and distribution calibration (44).

3 Methodology

3.1 Problem Definition

Long-Tailed Learning. Let $D_{tr} = \{(x_i, y_i)\}_{i=1}^{N}$ be an imbalanced dataset with N samples and C classes, where x_i denotes a sample and $y_i = \mathcal{Y} \in \{1, \cdots, C\}$ denotes its label. Let n_C denote the number of instances within class c. Without loss of generality, we assume that classes are sorted by decreasing cardinality, i.e. $n_1 \geq n_2 \geq \cdots \geq n_C$. Furthermore, we define the *imbalance factor* (IF), ρ, of a dataset as $\rho = n_1/n_C$. A greater degree of imbalance typically results in poorer performance on tail classes.

We train a classification model $\phi(\cdot)$ consisting of two parts: a backbone network $f(x;\theta)$ that extracts a feature representation, $z_i \in \mathbb{R}^D$, and a classifier $g(z_i; W) \in \mathbb{R}^{C \times D}$ that outputs class predictions. Here, D is the feature dimension, and θ and W are the parameters for the backbone and classifier respectively. We omit the bias term of the linear classifier for brevity unless specified otherwise. As we expect consistent performance across all classes, we evaluate our model on a class-balanced test distribution (i.e. $\rho = 1$).

ERM. Given a loss function $\mathcal{L}$, the objective of empirical risk minimization (ERM) can be expressed as:

$$\min_{\theta} \mathbb{E}_{(x,y)\sim P_\delta}\{\mathcal{L}(f(x_i;\theta), y_i)\} \tag{1}$$

in which $P_\delta = \frac{1}{N}\sum_{i=1}^{N} \delta(X = x_i, Y = y_i)$ is an estimate of the true data distribution, $P(X, Y)$. Unfortunately, this approximation makes the naive assumption that the training distribution accurately reflects the test distribution (57).

3.2 Analysis of mixup

Based on Vicinal Risk Minimization (VRM), mixup synthesizes new training samples by linearly interpolating pairs of examples and their labels. More specifically, given two random examples from the training data, (x_i, y_i) and (x_j, y_j), mixup combines them using 2:

$$\begin{aligned} \tilde{x} &= \lambda \cdot x_i + (1 - \lambda) \cdot x_j \\ \tilde{y} &= \lambda \cdot y_i + (1 - \lambda) \cdot y_j \end{aligned} \tag{2}$$

where y_i, y_j are one-hot vectors, $\lambda \sim Beta(\alpha, \alpha)$, $\alpha \in (0, \infty)$. When applied to all the data, 2 constructs a new dataset $D_v = \{(\tilde{x_i}, \tilde{y_i}\}_{i=1}^N$ that approximates the true distribution with the empirical vicinal distribution, $P_v(\tilde{x}, \tilde{y}) = \frac{1}{N} \sum_{i=1}^N v(\tilde{x}, y | x_i, y_i)$. Thus, the new objective from VRM becomes:

$$\min_\theta \mathbb{E}_{(\tilde{x}, \tilde{y}) \sim P_v} \{\mathcal{L}(f(\tilde{x_i}; \theta), \tilde{y_i})\} \tag{3}$$

Previous works have shown that mixup training improves calibration (37; 58), robustness (55), and generalization of networks (55; 8) in both balanced and imbalanced settings. In the latter case, when coupled with instance-balanced (natural) sampling (23), mixup demonstrates the ability to promote more *balanced* weight norms (58). To better understand the effect of mixup on representation learning, we first divide the mixup data into two classes: (1) Mixing between samples of the same class, and (2) Mixing between samples of different classes (i.e. head and tail). We then attempt to understand the effect of both cases on intra-class variance.

Lemma 1(a) (Input variance of interpolated samples within the same class)**.**

Let x_0 and x_1 be two random variables drawn i.i.d from the same class conditional distribution $X \sim \mathcal{N}(\mu_x, \sigma_x^2)$. Let $\tilde{x}$ denote the mixed sample, where $\tilde{x} = \lambda_0 x_0 + \lambda_1 x_1$ and $\lambda_1 = 1 - \lambda_0$. Then, the variance of $\tilde{x}$ is less than that of $\tilde{x}$ by a factor of $\lambda^2 + (1 - \lambda)^2$.

Proof Sketch 1(a). Let $\sigma_{\tilde{x}}^2$ be the variance of the mixed sample $\tilde{x}$. When x_0 and x_1 have the same label, we get:

$$\begin{aligned}
\sigma_{\tilde{x}}^2 &= \mathbb{E}[(\tilde{x} - \mu_{\tilde{x}})^2] \\
&= \mathbb{E}[(\sum_{i=0}^{1} \lambda_i x_i - \sum_{i=0} \lambda_i \mu_X)^2] \\
&= \mathbb{E}[(\sum_{i=0}^{1} \lambda_i (x_i - \mu_X))^2] \\
&= \mathbb{E}[\sum_{j=0}^{1} \sum_{i=0}^{1} a_j a_i (x_i - \mu_X)(x_j - \mu_X)] \\
&= \sum_{i=0}^{1} \lambda_i^2 \mathbb{E}[(x_0 - \mu_X)^2] \mathbb{E}[x_1 - \mu_X)^2] + 2\lambda_0 \lambda_1 \mathbb{E}[(x_0 - \mu_X)(x_1 - \mu_X)] \\
&= \sum_{i=0}^{1} \lambda_i^2 \sigma_X^2 + 2\lambda_0 \lambda_1 \mathbb{E}[(x_0 - \mu_X)(x_1 - \mu_X)]
\end{aligned}$$

Since $\mathbb{E}[(x_0 - \mu_X)(x_1 - \mu_X)]$ is the covariance of two independent variables, the second term becomes 0. Thus, we get:

$$\sigma_{\tilde{x}}^2 = (\lambda_0^2 + \lambda_1^2)\sigma_X^2 = (\lambda_0^2 + (1 - \lambda_0)^2)\sigma_X^2 \leq \sigma_X^2$$

where in the last line, we use the fact that λ is drawn from a *Beta* distribution, and hence $\lambda_0^2 + \lambda_1^2 = \lambda_0^2 + (1 - \lambda_0^2) \leq 1$.

Lemma 1(b) (Input variance of interpolated samples between different classes)**.**

Given a long-tailed dataset $\{(x_i, y_i)\}_{i=1}^n$*, let* (x_0, y_0) *and* (x_1, y_1) *be two random samples drawn i.i.d from a two different classes respectively. Then, the variance of the mixed sample* $\tilde{x} = \lambda_0 x_0 + \lambda_1 x_1$ *is bounded by* $(\sigma_{x_0}^2, \sigma_{x_1}^2)$.

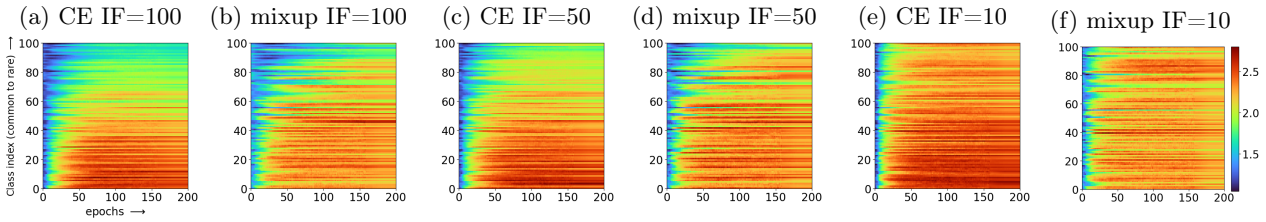

Figure 3: When training a model with just CE (see **(a)**, **(c)**, **(e)**), the weights of the head classes grow at a rate much faster than that of the tail classes. This is somewhat ameliorated with mixup, as shown in **(b)**, **(d)**, and **(f)**. The "balancing" effect of mixup is more prominant as the imbalance factor decreases.

Proof Sketch 1(b). We begin in the same way as the previous proof, the difference being that our samples have different class means.

$$
\begin{aligned}
\sigma_{\tilde{x}}^2 &= \mathbb{E}[(\sum_{i=0}^{1} \lambda_i x_i - \sum_{i=0}^{1} \lambda_i \mu_i)^2] \\
&= \mathbb{E}[(\sum_{i=0}^{1} \lambda_i (x_i - \mu_i))^2] \\
&= \mathbb{E}[\sum_{j=0}^{1} \sum_{i=0}^{1} \lambda_j \lambda_i (x_i - \mu_i)(x_j - \mu_j)] \\
&= \sum_{j=0}^{1} \sum_{i=0}^{1} a_i a_j \mathbb{E}[(x_i - \mu_i)(x_j - \mu_j)] \\
&= \lambda_0^2 \sigma_{x_0}^2 + \lambda_1^2 \sigma_{x_1}^2 - 2\sigma_{x_0}\sigma_{x_1}\mathbb{E}[(x_0 - \mu_{x_0})(x_1 - \mu_{x_1})] \\
&= \lambda_0^2 \sigma_{x_0}^2 + \lambda_1^2 \sigma_{x_1}^2 \\
&= \lambda_0^2 \sigma_{x_0}^2 + (1 - \lambda_0)^2 \sigma_{x_1}^2
\end{aligned}
$$

Since mixup finds linear interpolations of the data, it follows that the variance of the mixed sample is bounded by the variances of its sample components. Consequently, mixup is beneficial for highly dispersed tail classes, as it encourages models to learn better tail representations with the help of the head. To substantiate our claim, we train a plain cross-entropy model with and without mixup. We illustrate the learning of tail and head class representations by visualizing classifier weight norms (see Fig.4a) and intra-class variances (see Fig.4b). The top-1 accuracies are listed in Table 1. From these results, we observe that mixup training only marginally improves the top-1 accuracy of the model compared to CE. Despite this, it encourages the classifier weight norms of the head and tail classes to be more similar, and substantially decreases the intra-class variance of tail classes. As such, the class-covariances also become more similar across classes, as illustrated in Fig. 4c and in Fig. 4d.

To further investigate how the classifier weight norms are being "balanced", we plot their growth

over the course of training in Fig. 3. Unsurprisingly, we find that the norms for the head classes grow much faster than those of the tail, especially when the imbalanced factor is high. However, we also note that the norms of the tail classes seem to increase at a much slower rate towards the latter half of training. This, combined with the fact that the mixed samples still follow a head-majority (49), motivates us to incorporate another data augmentation method into training. Given T_2 total epochs, we propose to train with mixup for T_1 epochs, and use ISDSA (46) for the other $T_2 - T_1$ epochs.

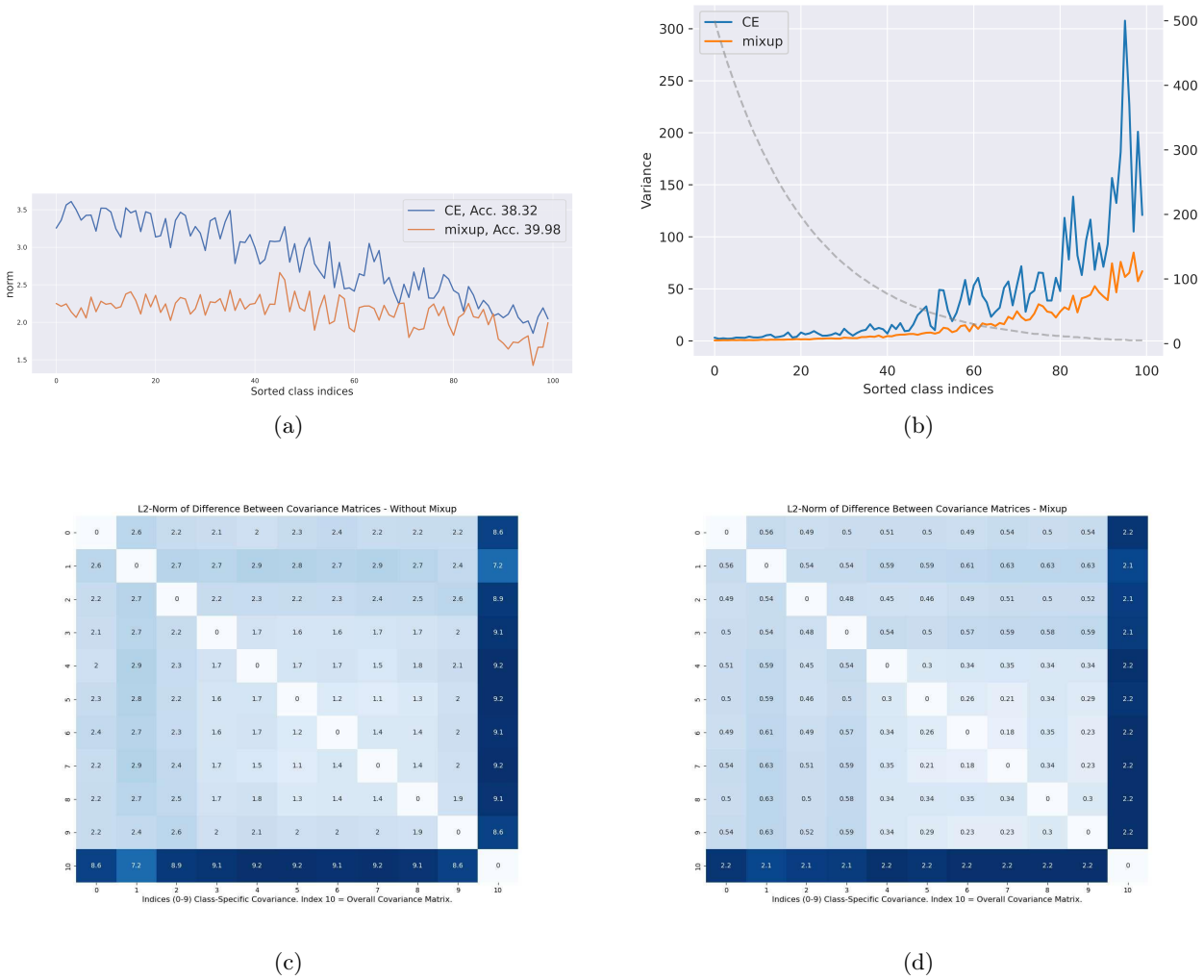

Figure 4: **(a)** Per-class weight norms for a model trained without mixup (*blue*) and with mixup (*orange*) on CIFAR100-LT. **(b)** The intra-class variances for each class given a model trained without mixup (*blue*) and with mixup (*orange*). **(c)** The L2 norms of the difference between per-class covariance matrices without mixup training and **(d)** with mixup training on CIFAR10-LT. The figures are best viewed in color.

3.3 Implicit Semantic Data Augmentation (ISDA)

Explicit data augmentation techniques, such as flipping or rotating an image, are commonly employed in training deep learning networks to improve a model's accuracy and generalization. However, these methods are limited in their ability to augment semantic information, which can be critical for increasing data diversity, especially for less common classes. Moreover, as the number of augmented samples used in training increases, the computational complexity of the training process also increases.

Fortunately, Wang *et al.* (46) proposes a solution to this problem through an implicit semantic data augmentation (ISDA) technique. The motivation of ISDA lies in the fact that deep features are typically linearized (39), allowing for the existence of numerous semantic directions to exist in feature space. When translating a feature along one of those directions, the feature is transformed semantically in a way that preserves class identity. For example, one meaningful direction might be to change the visual angle of an object in an image.

Specifically, for any given feature z_i, ISDA samples semantic directions from a zero-mean normal distribution, $\mathcal{N}(0, \Sigma_{y_i})$ to augment the feature with. Here, Σ_{y_i} is the class-wise covariance matrix corresponding to class y_i, and is estimated in an online fashion, outlined in Section 3.3.1. Since it is computationally exhaustive to explore all possible meaningful directions within the distribution, ISDA computes an upper bound on the CE loss that considers augmenting z_i in infinite directions:

$$\begin{aligned}\mathcal{L}_{ISDA} &= \sum_{i=1}^{N} L_{\infty}(f(x_i;\theta), y_i; \Sigma) \\ &= \sum_{i=1}^{N} -log(\frac{e^{(w_{y_i}^T z_i + b_{y_i})}}{\sum_{j=1}^{C} e^{(w_j^T z_i + b_j) + \frac{\lambda}{2}(w_j^T - w_{y_i}^T)\Sigma_{y_i}(w_j - w_{y_i})}})\end{aligned} \tag{4}$$

From 4, we can observe that ISDA is highly dependent on the training data. To ensure that the head classes will not dominate the augmentation method, we follow (13; 24) and re-weigh Eq. 4 by the effective number of samples $E_c \approx (1-\beta)/(1-\beta^{N_c})$, where $\beta = (N-1)/N$. Thus, our final loss can be written as:

$$\begin{aligned}\mathcal{L}_{ISDA} &= \sum_{i=1}^{N} E_i L_{\infty}(f(x_i;\theta), y_i; \Sigma) \\ &= \sum_{i=1}^{N} -log(\frac{e^{(w_{y_i}^T z_i + b_{y_i})}}{\sum_{j=1}^{C} e^{(w_j^T z_i + b_j) + \frac{\lambda}{2}(w_j^T - w_{y_i}^T)\Sigma_{y_i}(w_j - w_{y_i})}})\end{aligned} \tag{5}$$

3.3.1 Online Covariance Estimation

In the original ISDA algorithm, the online covariance estimation is performed as follows: at any t^{th} mini-batch during training, let m_j^t as the number of training samples in the $j-th$ class in the mini-batch. Let n_j^t as the number of training samples in the $j-th$ class in all t mini-batches, and is estimated as the sum of the previous estimation, $n_j^{(t-1)}$ and m_j^t, or $n_j^t = n_j^{(t-1)} + m_j^t$. Then, the covariance can be computed using 7.

$$\mu_{\mu_j}^t = \frac{n_j^{(t-1)}\mu_j^{(t-1)} + m_j^t \mu'^t_j}{n_j^{(t-1)} + m_j^t} \tag{6}$$

$$\Sigma_j^t = \frac{n_j^{(t-1)}\Sigma_j^{(t-1)} + m_j^t \bar{\Sigma}_j^t}{n_j^{(t-1)} + m_j^t} + \frac{n_j^{(t-1)} m_j^t (\mu_j^{(t-1} - \bar{\mu}_j^t)(\mu_j^{(t-1)} - \bar{\mu}_j^t)^T}{(n_j^{(t-1)} + m_j^t)^2} \tag{7}$$

where μ_j^t and Σ_j^t are the means and covariances of j^{th} class' features at t^{th} step, and $\bar{\mu_j}^t$ and $\bar{\Sigma_j}^t$ are the means and covariances of j^{th} class in the t^{th} mini-batch.

While ISDA is an effective augmentation strategy for balanced datasets, it is limited in its ability to learn appropriate covariance matrix for tail classes in long-tailed datasets due to its sparsity. MetaSAug (24) proposes a balanced meta-dataset to adjust the estimated class-wise covariance. However, the covariance estimation may still be inadequate, as its computation remains dependant on a small number of data. We mitigate the problem of having sparse covariance matrices by augmenting samples trained with mixup, which forces them to be full rank. To this end, we propose adding an additional *similarity loss* to ISDA that pushes the covariance matrices $\Sigma = \{\Sigma_1, \Sigma_2, \cdots, \Sigma_C\}$ to their collective mean.

$$\mathcal{L}_{sim} = ||\sum_{k}^{C} \Sigma_k - \frac{1}{C}\sum_{k=1}^{C} \Sigma_k|| \tag{8}$$

We can then update the class-wise covariance Σ using the gradient of Eq. 8:

$$\Sigma \leftarrow \Sigma - \gamma \nabla \mathcal{L}_{sim} \tag{9}$$

where γ is the gradient step size for Σ. In turn, the updated covariance is used to calculate the loss in Eq. 4. We postulate that this similarity loss not only provides a more robust estimate for the tail class covariances, but augments them with more diverse semantic transformations from the head classes. The pseudo-code for our representation learning algorithm is presented in 1.

Algorithm 1 Learning algorithm for Stage 1 Training

Input: Training set $D = \{(x_i, y_i)\}_{i=1}^{N}$, ending steps T_1, T_2

for $t \leq T_1$ **do**
  Sample a mini-batch $B \sim D$
  for each example $(x_i, y_i) \in B$ **do**
    Sample (x_j, y_j) from $P(\cdot \mid (x_i, y_i))$ and $\lambda \sim Beta(\alpha, \alpha)$
    $\tilde{x} \leftarrow \lambda x_i + (1 - \lambda) x_j$
    $\tilde{y} \leftarrow \lambda y_i + (1 - \lambda) y_j$ ▷ one-hot vectors
    Compute $\mathcal{L}_{mixup} = \mathcal{L}_{ce}(f(\tilde{x}; \Theta), \tilde{y})$
    Update model with SGD
for $T_1 \leq t < T_2$ **do**
  Sample a mini-batch $B \sim D$
  Estimate covariance matrices Σ_k per class $k \in C$
  Compute $\mathcal{L}_{sim} = || \sum_{k}^{C} \Sigma_k - \frac{1}{C} \sum_{k=1}^{C} \Sigma_k ||$ ▷ Push Σ's to mean
  Update $\Sigma \leftarrow \Sigma - \gamma \nabla \mathcal{L}_{sim}$
  Compute $\mathcal{L}_B$ with updated Σ:
  $L_B = \sum_{i=1}^{|B|} \epsilon_i \mathcal{L}_{ISDA}(f(x_i; \Theta), y_i; \Sigma)$
  Update model with SGD

3.4 Classification with Linear Discriminant Analysis (LDA)

3.4.1 Theoretical Motivation

Despite its popular usage on balanced datasets, the Softmax classifier is limited by its bias towards high class frequencies on long-tailed datasets. In general, the Softmax classifier on $P_{tr}(X, Y)$ can be modelled as:

$$\begin{aligned} \hat{y}_{tr} = \arg\max_{y_i \in \mathcal{Y}} P(Y = y_i | X = x) &= \arg\max_{y_i \in \mathcal{Y}} \frac{P(X = x | Y = y_i) P(Y = y_i)}{P(X)} \\ &\propto \arg\max_{y \in \mathcal{Y}} P(X = x | Y = y_i) P(Y = y_i) \end{aligned} \tag{10}$$

with $\hat{y_{tr}}$ being the class prediction and y_i is the label of interest. Despite the assumption that both training and testing instances come from the same underlying distribution $P(X|Y = y_i)$, their priors $P(Y = y_i)$ and evidence factors $P(X)$ differ. As such, their posteriors will also be different. This becomes more apparent when estimating the posterior of the test distribution:

$$\hat{y}_{te} = \arg\max_{y_i \in \mathcal{Y}} P_{te}(Y = y_i | X = x) \propto \arg\max_{y_i \in \mathcal{Y}} \frac{P_{te}(Y = y_i)}{P_{tr}(Y = y_i)} \cdot \frac{P_{tr}(Y = y_i | X = x)}{P_{tr}(Y = y_i)} \tag{11}$$

in which the difference between a long-tailed training distribution and a uniform test distribution in $\frac{P_{te}(Y = y_i)}{P_{tr}(Y = y_i)}$ creates a biased estimate of the test distribution.

3.4.2 Linear Discriminant Analaysis

Linear Discriminant Analysis (LDA) is a classical statistical learning method that models input class distributions as class-conditional Gaussian distributions with a tied covariance matrix. Under this modelling assumption, the optimal decision boundary is linear, and samples are classified by selecting the class cluster with the highest posterior probability. Considering that our initial training approach aims to make the class covariances more comparable, we believe that LDA (Linear Discriminant Analysis) be more advantageous in our scenario than Softmax.

LDA is commonly formulated using Fisher's LDA, which aims to find a lower-dimensional subspace that maximizes the separation between classes by minimizing within-class variance and maximizing between-class variance. Formally, let $U \in \mathbb{R}^{D \times P}, P > 1$ be a P-dimensional subspace with the most discriminative power. Let $X = \{x_1, \cdots, x_N\} \in \mathbb{R}^{d \times N}$ be the input data. The Fisher criterion is provided by:

$$J(U) = \frac{Tr(U^T S_B U)}{Tr(U^T S_W U)} \tag{12}$$

which we want to maximize with respect to U. Here, S_B is the *between-class scatter* matrix while S_W is the *within-class scatter* matrix. To account for the class-imbalance, we propose to normalize S_W by the number of samples N_c in each class $c \in \{1, \ldots, C\}$:

$$S_W = \sum_{c=1}^{C} \frac{1}{N_c} \sum_{i=1}^{N_c} (x_i - \mu_c)(x_i - \mu_c)^T, \quad \text{where} \tag{13}$$

$$\mu_c = \frac{1}{N_c} \sum_{i=1}^{N_c} n_i^{(c)} \tag{14}$$

Following Variant 2 in (19), we determine the between-scatter S_B by first defining the covariance matrix of the entire data, or the *total scatter* matrix S_T (Eq.15). Utilizing the property that $S_T = S_W + S_B$, the derivation of S_B is obtained in Eq. 16:

$$S_T = \frac{1}{N} \sum_{i=1}^{N} (x_i - \mu)(x_i - \mu)^T, \text{ where } \mu = \frac{1}{\sum_{c=1}^{C} N_c} \sum_{i=1}^{C} n_i \tag{15}$$

$$S_B = S_T - S_W = \sum_{c=1}^{C} (\mu_c - \mu)(\mu_c - \mu)^T \tag{16}$$

Optimization of Eq. 12 has been well-studied in the past (18; 4; 47) as a generalized eigenvalue

problem (S_B, S_W). We use the following formulation:

$$S_B U = S_W U \Lambda \Longrightarrow U = eig(S_W^{-1} S_B) \tag{17}$$

Here, U has columns corresponding to eigenvectors stacked column-wise from largest to smallest, while the entries of the diagonal of $\Lambda \in \mathbb{R}^{D \times D}$ are the corresponding eigenvalues. We defer readers to (18) for a more comprehensive overview of the optimization problem. To alleviate potential numerical instability (S_W can be invertible), we employed the use of several tricks, including the addition of a small positive constant, ϵ to the diagonal of S_W, and Robust FDA (15; 20). We find the former to be sufficient with less computation overhead. Thus, Eq. 17 can be rewritten as:

$$U = eig((S_W + \varepsilon I)^{-1} S_B) \tag{18}$$

3.4.3 Classification Rule

To evaluate our model, we employ LDA to derive the linear classifier weights $W \in \mathbb{R}^{D \times C}$. Assuming the input data Z is a class-conditional Gaussian distribution with class priors $\pi \in \Delta(C)$, mean $\mu \in \mathbb{R}^{(C \times D)}$, and a shared covariance matrix $\Sigma \in \mathcal{R}^{D \times D}$, the LDA classifier can be interpreted as selecting the class-conditional Gaussian with the highest posterior probability. Let the bias parameter be denoted as W^0. The classifier weight and bias terms are as follows:

$$W^0 = \frac{-1}{2} diag(\mu \Sigma^{-1} \mu^T) + \ln \pi \tag{19}$$

$$W = \Sigma^{-1} \mu^T \tag{20}$$

Finally, the classification function is expressed as:

$$g_W(z) = W^0 + W^T Z \tag{21}$$

A full visualization of our method can be found in Appendix A.

4 Experiments

We carry out extensive extensive experiments to demonstrate the efficacy of our framework on long-tailed recognition. In Section 4.1, we introduce the datasets used for training and testing, as well as the implementation details of our method. Then, in Section 4.2, we ablate the design choices of our method with the factors detailed in Sections 3.2, 3.3, and 3.4. Finally, we compare the results of our best-performing method with current state-of-the-art LTR methods.

4.1 Experiment Setup

4.1.1 Datasets

CIFAR-10/100 LT Both CIFAR-10 and CIFAR-100 datasets consist of 60,000 images, with 50,000 images designated for training and 10,000 images for validation. Images in CIFAR-10 are divided into 10 classes, whereas those in CIFAR-100 are divided into 100 classes. For LTR, we sample a long-tailed versions of the CIFAR datasets with the same settings as those used in (7). Following the experiments conducted by (7) and (59), we perform experiments with imbalance factors 100, 50, and 10.

ImageNet-LT ImageNet-LT is a long-tailed version of the large-scale object classification dataset ImageNet (14). It is obtained by sampling a subset based on the Pareto distribution with a power value of $\alpha = 6$. This dataset contains 115.8K images from 1,000 categories, with class cardinalities ranging from 5 to 1,280.

iNaturalist 2018 The iNaturalist 2018 dataset (41) is a large, fine-grained species classification dataset with 437.5K images across 8,142 categories. Out of the four datasets we use, iNaturalist is the only one that is naturally imbalanced.

4.1.2 Implementation Details

We follow Kang *et al.* (23) in decoupling representation and classifier learning. To ensure a fair comparison with previous studies, we adopt a ResNet-32 as our backbone architecture for CIFAR-10 and 100, and ResNet-50 for Imagenet and iNaturalist. We also train our models in a deterministic fashion with a seed of 0.

Stage 1: Representation Learning For CIFAR10-LT and CIFAR100-LT, we train a ResNet-32 using stochastic gradient descent (SGD) with momentum 0.9, learning rate 0.01, a weight decay of $2e^{-4}$, and a multistep learning rate schedule that decays by 0.01 at epochs 160 and 180. Models for the CIFAR datasets are trained on a single GPU with a batch size of 100 for 200 epochs. For Imagenet-LT and iNaturalist, we alter the learning rate schedule to decay by 0.1 at epochs 60 and 80, and train our models on four GPUs with a batch size of 64. The optimal value for the hyperparameter γ is found using grid search on the values $\{0.0, 0.25, 0.50, 0.75, 1.0\}$. We train all models according to Algorithm 1 with $T_1 = 160$, and employ the use of both instance-balanced sampling (23) and an auxiliary softmax classifier. This classifier is discarded at the end of training. In addition to the augmentations proposed in 3, we utilize random cropping and flipping.

Stage 2: Classifier Learning At the end of Stage 1, we keep our backbone parameters constant and fit LDA using the fixed training features. During inference, we substitute the auxiliary softmax classifier for the LDA classifier in Eq. 21. Note that as we do not use the LDA classifier during training, we can omit this step up until the end of training to reduce the overhead of our method.

4.2 Ablation Study

We study (1) the addition of ISDA to our Stage 1 pipeline, (2) the addition of our similarity loss to ISDA, and (3) the use of a LDA classifier on features trained with and without mixup, ISDA, and ISDA with our similarity loss. The results of our ablations are presented in Table 1.

Table 1: Ablation study on CIFAR10/100-LT of top-1 accuracy of different methods.

	(a) CIFAR10-LT			(b) CIFAR100-LT			(c) ImageNet-LT	(d) iNaturalist
Imbalance factor (ρ)	100	50	10	100	50	10	—	—
naive (CE)	71.40	77.56	87.01	38.32	43.15	56.90	41.6	61.7
+ LDA	73.43	78.29	87.24	39.41	45.00	55.86	42.3	63.4
mixup (53)	72.81	79.41	87.76	40.45	44.20	57.06	45.5	66.9
+ LDA	82.00	83.67	**89.22**	46.37	49.21	59.96	48.9	68.5
+ ISDA (46)	80.89	84.35	88.53	48.70	53.36	61.56	50.3	69.1
+ ISDA + LDA	82.17	84.84	88.91	45.68	53.93	61.30	49.2	67.4
+ ISDA + SimLoss	81.26	84.60	88.96	50.79	**53.99**	62.01	51.0	69.6
+ ISDA + SimLoss + LDA	**82.78**	**85.46**	89.02	**51.69**	53.52	**62.31**	**51.7**	**70.0**

Addition of ISDA. We tested the values $\{140, 160, 180\}$ for T_1 and found $T_1 = 160$ to work best. From Table 1, we can see that the incorporation of ISDA substantially improves the performance of mixup across all datasets, except CIFAR10-LT. For example, with CIFAR100-LT IF100, ISDA improves the top-1 accuracy of the model from 40.45% with mixup, to 53.93%. to Since CIFAR10-LT is relatively small, with only 10 classes, the data may be too limited for effective semantic transformations.

Addition of Similarity Loss. The addition of the similarity loss marginally improves upon the ISDA loss, supporting our reasoning in Section 3.3.1. However, more experiments will need to be carried out to ensure that this improvement is not attributed to noise. From Fig. 5, we observe that the intra-class variance and classifier weight norms for our method are significantly lower than if we were to use mixup. The exception is when the dataset is more balanced (i.e. IF=10), in which our stage 1 training increases the spread of the tail features to ensure that they occupy relatively the same "space" in feature space as more common classes. In addition, we show the classifier weight norms produced by our method in Fig. 6. Across all imbalance factors 100, 50, and 10, our method produces lower and more balanced norms.

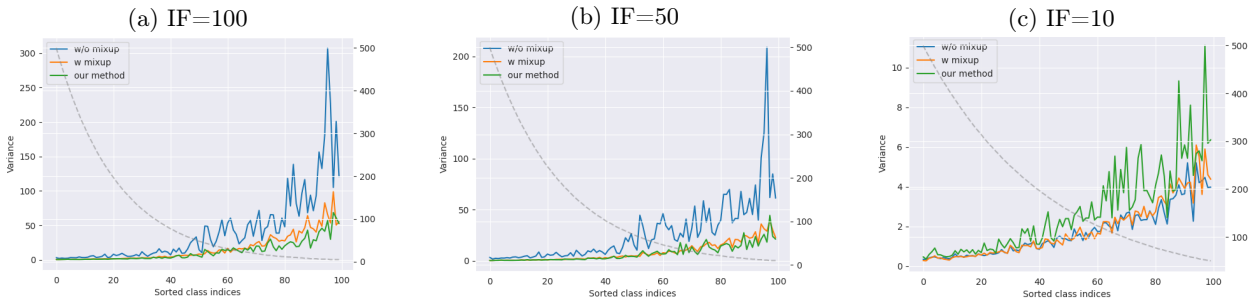

Figure 5: Comparison of the intra-class variances of a model trained with our method (*green*) versus with mixup (*(orange)*) and without (*blue*) on CIFAR100-LT. The left y-axis represents the scale, while the right y-axis denotes the number of samples in a class.

LDA Classification. Unsurprisingly, when classifying features produced from a model trained with CE, the LDA classifier only performs marginally better than a Softmax classifier. One reason for this is that the class covariances largely differ from each other, as seen in Fig. 4b. Additionally, the sample size of some tail classes is significantly less than its dimensionality, which poses an issue for LDA. However, the top-1 accuracy is substantially improved when using a model trained with mixup. This further substantiates our claim that mixup promotes more similar class covariances. Interestingly, the amount of improvement decreases as the dataset becomes more "balanced", i.e. lower imbalance factor.

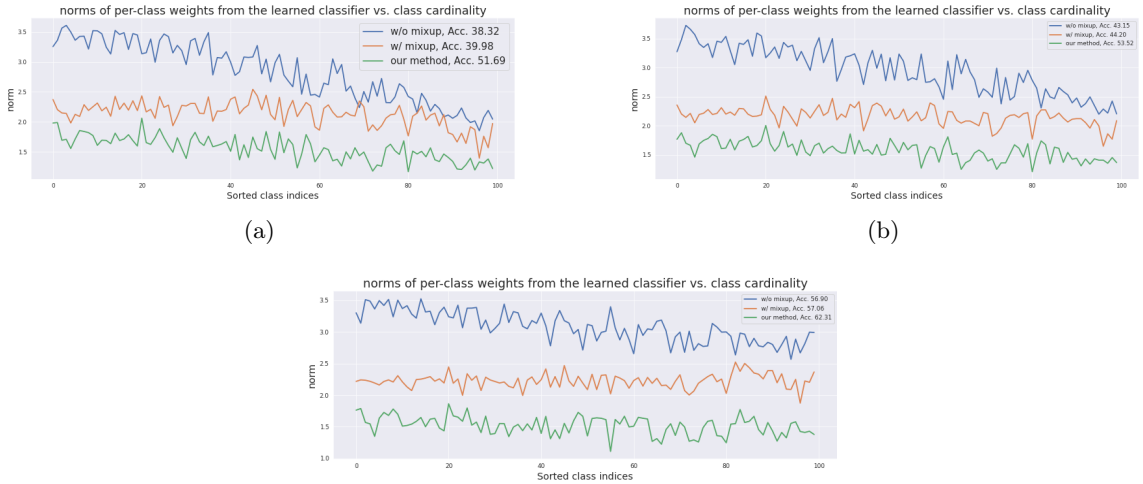

Figure 6: Comparison of the classifier weight norms of a model trained with our method (*green*) versus with mixup (*(orange)*) and without (*blue*) on CIFAR100-LT.

For features produced from a model trained with ISDA, the LDA classifier performs slightly better than the Softmax classifier. LDA also demonstrates improvement over Softmax when the similarity loss is incorporated into training.

4.3 Benchmark Results

Methods for Comparison. We compare the performance of our framework with the most relevant methods in the field. We select methods that are a blend of different techniques, such as CE+CB (13) for loss re-weighting, BCL (59) for supervised contrastive learning, as well as methods that draw some similarities to ours, including MiSLAS (58) and (24). Table 2 shows the benchmark results for CIFAR10-LT and CIFAR100-LT, whereas Table 3 shows the results for Imagenet-LT and iNaturalist.

Results. Despite the simplicity of our framework, we are able to outperform many of the methods listed in Table 2 and 3. Across all datasets, our method is able to reach near state-of-the-art results without bells and whistles, falling slightly behind BCL (59) and PaCo (12), which rely on contrastive learning and are far more computationally intensive.

Table 2: Top-1 Accuracy on Benchmark Datasets

	(a) CIFAR10-LT			(b) CIFAR100-LT		
Imbalance factor (ρ)	100	50	10	100	50	10
naive (CE)	71.40	77.56	87.01	38.32	43.85	55.71
CE+CB (13)	74.57	79.27	87.49	39.60	45.32	57.99
LogitAdj (29)	80.92	—	—	42.01	47.03	57.74
LDAM-DRW (7)	77.03	81.03	88.16	42.04	46.62	58.71
τ-norm (23)	—	—	—	47.73	52.53	63.80
RIDE (3-expert) (45)	—	—	—	48.60	51.40	59.80
MiSLAS (58)	82.12	85.71	90.00	47.02	52.33	63.25
DRO-LT (32)	—	—	—	47.31	57.57	63.41
MetaSAug-LDAM (24)	80.66	84.34	89.68	48.01	52.27	61.28
BCL (59)	**84.32**	**87.24**	**91.12**	51.93	**56.59**	**64.87**
PaCo (12)	—	—	—	**52.00**	56.00	64.20
Ours	82.78	85.46	89.02	51.69	53.99	62.31

Table 3: Top-1 Accuracy on Benchmark Datasets

	(c) ImageNet-LT	(d) iNaturalist
Imbalance factor (ρ)	—	—
naive (CE)	44.4	61.7
CE+CB (13)	33.2	54.0
LogitAdj (29)	51.1	68.4
LDAM-DRW (7)	49.8	66.1
τ-norm (23)	49.4	65.6
RIDE (3-expert) (45)	54.9	72.2
MiSLAS (58)	52.7	71.6
DRO-LT (32)	53.5	69.7
MetaSAug-LDAM (24)	47.4	68.8
BCL (59)	**56.0**	71.8
PaCo*	54.4	**72.3**
Ours	51.7	70.0

5 Conclusion

In this major qualifying project, we explored the task of long-tail image classification, a crucial problem in real-world vision applications.

The problem began with an extensive literature review of previous LTR methods, which continued throughout the entire project as new publications became relevant. We initially began the project by developing a theoretical understanding for why certain representation learning methods were successful. Once we had sufficient knowledge of this area, we extended this theory to classifier learning and sought to unite the two. A large portion of this project became centered around the empirical analysis, which became exhaustive

due to all the different ideas, baseline methods, hyperparameters, and benchmark scenarios we wanted to implement. Overall, we are pleased with the knowledge and experience we gained from this MQP. We believe our proposed method is a strong baseline and perspective for future work to build upon. We provide a detailed discussion on future directions below.

5.1 Future Work

Transfer Learning between the Head and Tail To better augment tail classes in ISDA, we can look at transferring semantic transformation directions from head classes. For example, given a common class "Cat" and a rare class "Caracal", which are both felines, we can augment the "Caracal" class by "adding stripes to the fur", which is also a semantic transformation direction available to the "Cat" class. Such an augmentation scheme would be more robust than augmenting all classes based on a shared covariance.

Multi-Domain Long-Tailed Learning. Up till now, previous LTR solutions operate on the crucial assumption that the conditional distributions of the training and test sets remain equal, despite the discrepancy between their label distributions, i.e., $P_{train}(y) \neq P_{test}(y), P_{train}(y|x) = P_{test}(y|x)$. However, this assumption may not hold in practice, as training data can be drawn from multiple domains. For instance, one can enrich training data for autonomous driving by combining urban scenes from various cities or through simulation software. In this case, we can assume that each city or simulation environment produces images with different underlying distributions.

In MDLT, we extend the problem to multiple domains, such that the overall data distribution is drawn from a set of K domains $\{\mathcal{D}\}_{k=1}^{K}$, and each domain is associated with a class-imbalanced dataset $\{(x_i, y_i, d)\}_{i=1}^{K}$. Following (1; 51), the training and testing domains, $\mathcal{D}_{tr}$ and $\mathcal{D}_{te}$, can be expressed as a mixture distribution over domain space $\mathcal{D}$, $P^{tr} = \sum_{d=1}^{\mathcal{D}} \eta_d^{tr} P_d^{tr}$ and $P^{tr} = \sum_{d=1}^{\mathcal{D}} \eta_d^{te} P_d^{te}$, where η_d^{tr} and η_d^{te} denote the mixture probabilities. The corresponding training and testing domains are $\mathcal{D}^{tr} = \{d \in \mathcal{D} | \eta_d^t r > 0\}$ and $\mathcal{D}^{te} = \{d \in \mathcal{D} | \eta_d^{te} > 0\}$. Each label distribution within $\mathcal{D}^{tr}$ follows a long-tail distribution, as formulated in the previous section. Furthermore, we consider the setting where the label distribution is imbalanced across domains as well.

(51) proposes two types of test distributions, namely subpopulation shift and domain shift. In subpopulation shift, the test distribution is class-balanced and domain-balanced, or $\mathcal{D}^{te} \subseteq \mathcal{D}^{tr}$ and $\{\eta_d^{te} = \frac{1}{|\mathcal{D}^{te}|} \mid \forall d \in \mathcal{D}^{te}\}$. In domain shift, the test and training domains are disjoint. In this work, we will address both types of distributions in testing.

In order to connect ERM to MDLT, we follow previous works (17) in decoupling the feature extractor f and the classifier h. This allows us to derive the following test error ϵ_{te} with the importance sampling trick, to which we want to minimize:

$$\begin{aligned}
\epsilon_{te} &= \mathbb{E}_{(x,y)\sim P_{D_{te}}(x,y)} \mathcal{L}(h \circ f_\theta(x), y) \\
&= \mathbb{E}_{(x,y)\sim P_{D_{tr}}(x,y)} \mathcal{L}(h \circ f_\theta(x), y) \, \frac{P_{D_{te}}(f_\theta(x), y)}{P_{D_{tr}}(f_\theta(x), y)} \\
&= \mathbb{E}_{(x,y)\sim P_{D_{tr}}(x,y)} \mathcal{L}(h \circ f_\theta(x), y) \, \frac{P_{D_{te}}(y)}{P_{D_{tr}}(y)} \frac{P_{D_{te}}(f_\theta(x)|y)}{P_{D_{tr}}(f_\theta(x)|y)} \\
&= \mathbb{E}_{(x,y)\sim P_{D_{tr}}(x,y)} \mathcal{L}(h \circ f_\theta(x), y) \, w_y \, \alpha_{(x,y)}
\end{aligned} \tag{22}$$

Equation 22 serves as a good motivation towards combining LTR and domain generalization techniques. In LTR, we assume $\alpha_{(x,y)} = 0$, as $P_{tr}(x|y) = P_{te}(x|y)$, whereas in DG, we assume w_y is equal across all classes, $P_{tr}(y) = P_{te}(y)$.

References

[1] Isabela Albuquerque, João Monteiro, Tiago H. Falk, and Ioannis Mitliagkas. Adversarial target-invariant representation learning for domain generalization. *CoRR*, abs/1911.00804, 2019.

[2] Shaden Alshammari, Yuxiong Wang, Deva Ramanan, and Shu Kong. Long-tailed recognition via weight balancing. In *CVPR*, 2022.

[3] Yuri Aurelio, Gustavo De Almeida, Cristiano Castro, and Antônio Braga. Learning from imbalanced data sets with weighted cross-entropy function. *Neural Processing Letters*, 50, 10 2019.

[4] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, USA, 2004.

[5] Mateusz Buda, Atsuto Maki, and Maciej A. Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Netw.*, 106(C):249–259, oct 2018.

[6] Jonathon Byrd and Zachary Lipton. What is the effect of importance weighting in deep learning? In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 872–881. PMLR, 09–15 Jun 2019.

[7] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[8] Luigi Carratino, Moustapha Cissé, Rodolphe Jenatton, and Jean-Philippe Vert. On mixup regularization, 2022.

[9] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. SMOTE: Synthetic minority oversampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, jun 2002.

[10] Hsin-Ping Chou, Shih-Chieh Chang, Jia-Yu Pan, Wei Wei, and Da-Cheng Juan. Remix: Rebalanced mixup, 2020.

[11] Ekin Dogus Cubuk, Barret Zoph, Jon Shlens, and Quoc Le. Randaugment: Practical automated data augmentation with a reduced search space. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 18613–18624. Curran Associates, Inc., 2020.

[12] 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.

[13] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge J. Belongie. Class-balanced loss based on effective number of samples. *CoRR*, abs/1901.05555, 2019.

[14] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[15] Weihong Deng, Jiani Hu, Jun Guo, and Honggang Zhang. Robust discriminant analysis of gabor feature for face recognition. In *Fourth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2007)*, volume 3, pages 248–252, 2007.

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

[17] Qi Dou, Daniel C. Castro, Konstantinos Kamnitsas, and Ben Glocker. Domain generalization via model-agnostic learning of semantic features. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[18] Benyamin Ghojogh, Fakhri Karray, and Mark Crowley. Eigenvalue and generalized eigenvalue problems: Tutorial, 2022.

[19] Benyamin Ghojogh, Fakhri Karray, and Mark Crowley. Fisher and kernel fisher discriminant analysis: Tutorial, 2022.

[20] Ming Guo and Zhelong Wang. A feature extraction method for human action recognition using body-worn inertial sensors. In *2015 IEEE 19th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pages 576–581, 2015.

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

[22] Nathalie Japkowicz and Shaju Stephen. The class imbalance problem: A systematic study. *Intell. Data Anal.*, 6(5):429–449, oct 2002.

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

[24] Shuang Li, Kaixiong Gong, Chi Harold Liu, Yulin Wang, Feng Qiao, and Xinjing Cheng. Metasaug: Meta semantic augmentation for long-tailed visual recognition. *CoRR*, abs/2103.12579, 2021.

[25] Tianhong Li, Peng Cao, Yuan Yuan, Lijie Fan, Yuzhe Yang, Rogerio Feris, Piotr Indyk, and Dina Katabi. Targeted supervised contrastive learning for long-tailed recognition. *arXiv preprint arXiv:2111.13998*, 2021.

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

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

[28] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X. Yu. Large-scale long-tailed recognition in an open world. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[29] Aditya Krishna Menon, Sadeep Jayasumana, Ankit Singh Rawat, Himanshu Jain, Andreas Veit, and Sanjiv Kumar. Long-tail learning via logit adjustment. *CoRR*, abs/2007.07314, 2020.

[30] Jiawei Ren, Cunjun Yu, Shunan Sheng, Xiao Ma, Haiyu Zhao, Shuai Yi, and Hongsheng Li. Balanced meta-softmax for long-tailed visual recognition, 2020.

[31] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning, 2019.

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

[33] Li Shen, Zhouchen Lin, and Qingming Huang. Relay backpropagation for effective learning of deep convolutional neural networks. In *ECCV*, 2016.

[34] Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting, 2019.

[35] Jingru Tan, Xin Lu, Gang Zhang, Changqing Yin, and Quanquan Li. Equalization loss v2: A new gradient balance approach for long-tailed object detection, 2021.

[36] Jingru Tan, Changbao Wang, Buyu Li, Quanquan Li, Wanli Ouyang, Changqing Yin, and Junjie Yan. Equalization loss for long-tailed object recognition, 2020.

[37] Sunil Thulasidasan, Gopinath Chennupati, Jeff Bilmes, Tanmoy Bhattacharya, and Sarah Michalak. On mixup training: Improved calibration and predictive uncertainty for deep neural networks, 2020.

[38] Sunil Thulasidasan, Gopinath Chennupati, Jeff A Bilmes, Tanmoy Bhattacharya, and Sarah Michalak. On mixup training: Improved calibration and predictive uncertainty for deep neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[39] Paul Upchurch, Jacob Gardner, Geoff Pleiss, Robert Pless, Noah Snavely, Kavita Bala, and Kilian Weinberger. Deep feature interpolation for image content changes, 2017.

[40] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[41] 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.

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

[43] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, Aaron Courville, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states, 2018.

[44] Chaozheng Wang, Shuzheng Gao, Cuiyun Gao, Pengyun Wang, Wenjie Pei, Lujia Pan, and Zenglin Xu. Label-aware distribution calibration for long-tailed classification. *arXiv preprint arXiv:2111.04901*, 2021.

[45] 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*, 2021.

[46] Yulin Wang, Xuran Pan, Shiji Song, Hong Zhang, Cheng Wu, and Gao Huang. Implicit semantic data augmentation for deep networks, 2020.

[47] Max Welling. Fisher linear discriminant analysis. Technical report, University of Toronto, 2005.

[48] David Wen, Saad M Khan, Antonio Ji Xu, Hussein Ibrahim, Luke Smith, Jose Caballero, Luis Zepeda, Carlos de Blas Perez, Alastair K Denniston, Xiaoxuan Liu, et al. Characteristics of publicly available skin cancer image datasets: a systematic review. *The Lancet Digital Health*, 4(1):e64–e74, 2022.

[49] Zhengzhuo Xu, Zenghao Chai, and Chun Yuan. Towards calibrated model for long-tailed visual recognition from prior perspective, 2021.

[50] Lu Yang, He Jiang, Qing Song, and Jun Guo. A survey on long-tailed visual recognition. *International*

Journal of Computer Vision, 130(7):1837–1872, may 2022.

[51] Xinyu Yang, Huaxiu Yao, Allan Zhou, and Chelsea Finn. Multi-domain long-tailed learning by augmenting disentangled representations, 2022.

[52] Yuhang Zang, Chen Huang, and Chen Change Loy. Fasa: Feature augmentation and sampling adaptation for long-tailed instance segmentation, 2021.

[53] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization, 2017.

[54] Linjun Zhang, Zhun Deng, Kenji Kawaguchi, Amirata Ghorbani, and James Zou. How does mixup help with robustness and generalization? In *International Conference on Learning Representations*, 2021.

[55] Linjun Zhang, Zhun Deng, Kenji Kawaguchi, Amirata Ghorbani, and James Zou. How does mixup help with robustness and generalization?, 2021.

[56] Linjun Zhang, Zhun Deng, Kenji Kawaguchi, and James Y. Zou. When and how mixup improves calibration. In *ICML*, 2022.

[57] Songyang Zhang, Zeming Li, Shipeng Yan, Xuming He, and Jian Sun. Distribution alignment: A unified framework for long-tail visual recognition. In *CVPR*, 2021.

[58] Shu Liu Zhisheng Zhong, Jiequan Cui and Jiaya Jia. Improving calibration for long-tailed recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[59] Jianggang Zhu, Zheng Wang, Jingjing Chen, Yi-Ping Phoebe Chen, and Yu-Gang Jiang. Balanced contrastive learning for long-tailed visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6908–6917, 2022.

Appendices

A Visualization of Proposed Framework

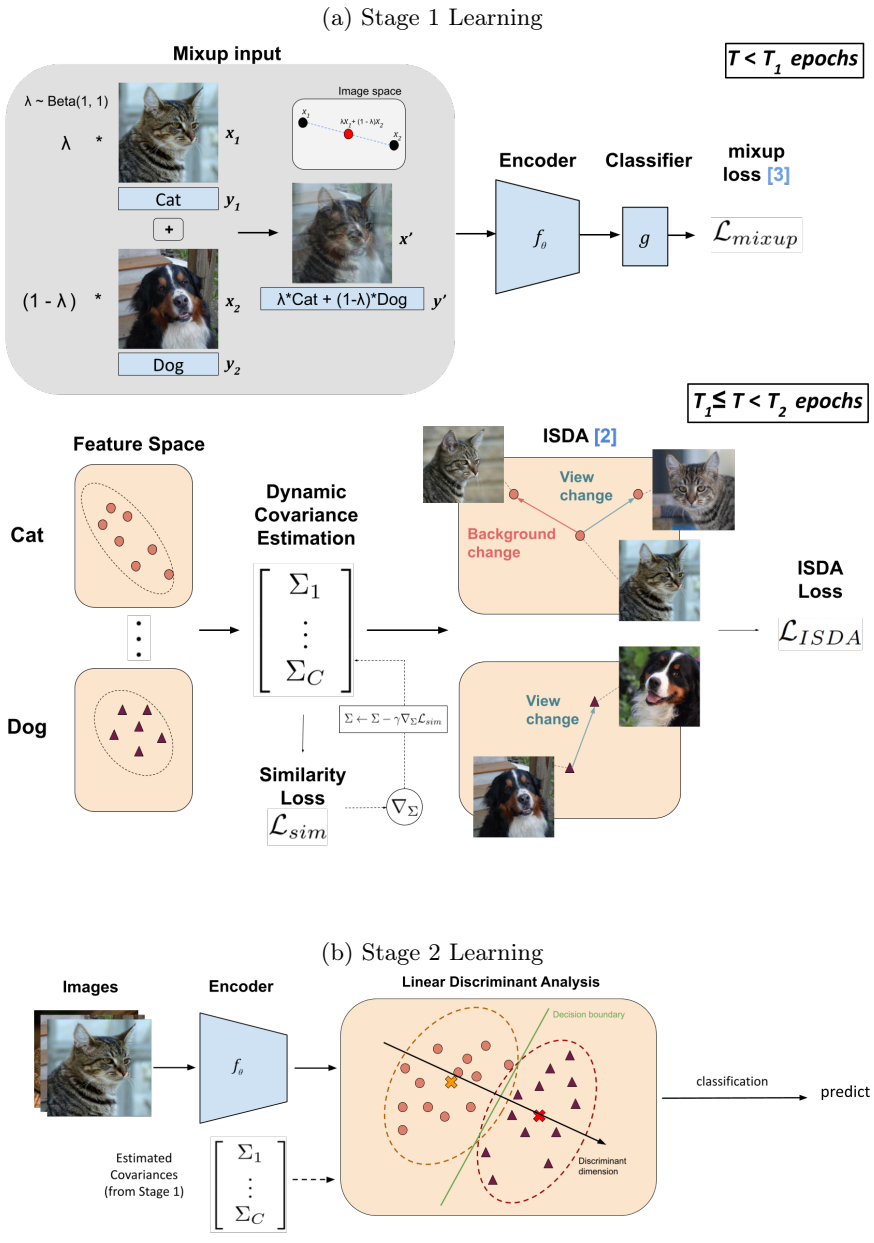

Figure 7: Illustration of our method, decoupled into **(a)** Representation Learning and **(b)** Classifier Learning