AutoBalance: Optimized Loss Functions for Imbalanced Data

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

Imbalanced datasets are commonplace in modern machine learning problems. The presence of under-represented classes or groups with sensitive attributes results in concerns about generalization and fairness. Such concerns are further exacerbated by the fact that large capacity deep nets can perfectly fit the training data and appear to achieve perfect accuracy and fairness during training, but perform poorly during test. To address these challenges, we propose AutoBalance, a bi-level optimization framework that automatically designs a training loss function to optimize a blend of accuracy and fairness-seeking objectives. Specifically, a lower-level problem trains the model weights, and an upper-level problem tunes the loss function by monitoring and optimizing the desired objective over the validation data. Our loss design enables personalized treatment for classes/groups by employing a parametric cross-entropy loss and individualized data augmentation schemes. We evaluate the benefits and transferability of our approach to the application scenarios of imbalanced and group-sensitive classification. Extensive empirical evaluations demonstrate the benefits of AutoBalance over state-of-the-art approaches. Our experimental findings are complemented with theoretical insights on loss function design and the benefits of the train-validation split.

Organization of Supplementary

1. Section A is the extended related works. Here, we provided a discussion for the related works of data augmentation and algorithm for Implicit Differentiation [49].
2. Section B provides the standard error of Table 1,2,3 and 4 over 5 runs as well as several extra experiments. We also extend the discussion of Fig. 3 in its caption.
3. Section C extends the discussion of Section 4 on the importance of validation in multi-objective learning and generalization.
4. Section D proves Lemma. 1 on the consistency of the parametric cross-entropy with multiplicative adjustments.
5. Finally, Section E establishes the benefit of data augmentation by relating "data-augmented cross-entropy" to parametric cross-entropy loss.

1 Introduction

Recently, deep learning, large datasets, and the evolution of computing power have led to unprecedented success in computer vision, and natural language processing [15, 41, 66]. This success is partially driven by the availability of high-quality datasets, built by carefully collecting a sufficient number of samples for each class. In practice, real-world datasets are frequently imbalanced and exhibit long-tailed behavior, necessitating a careful treatment of the minorities [21, 56, 24]. Indeed,

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

modern classification tasks can involve thousands of classes, so it is perhaps intuitive that some classes should be over/under-represented compared to others. Besides class imbalance, minorities can also appear at the feature-level; for instance, the specific values of the features of an example can vary depending on that example's membership in certain sensitive or protected groups, e.g. race, gender, disabilities (see also Figure 1a). In scenarios where imbalances are induced by heterogeneous client datasets (e.g., in the context of federated learning), addressing these imbalances can help ensure that a machine learning model works well for all clients, rather than just those that generate the majority of the training data. This rich set of applications motivate the careful treatment of imbalanced datasets.

In the imbalanced classification literature, the recurring theme is maximizing a fairness-seeking objective, such as balanced accuracy. Unlike standard accuracy, which can be dominated by the majorities, a fairness-seeking objective seeks to promote examples from minorities, and downweigh examples from majorities. Here, note that there is a distinction between the test and training objectives. While the overall goal is typically to maximize a non-differentiable objective such as balanced accuracy on the test set, during training, we use a differentiable proxy for this, such as weighted cross-entropy. Thus, the fundamental question of interest is:

How to design a training loss to maximize a fairness-seeking objective on the test set?

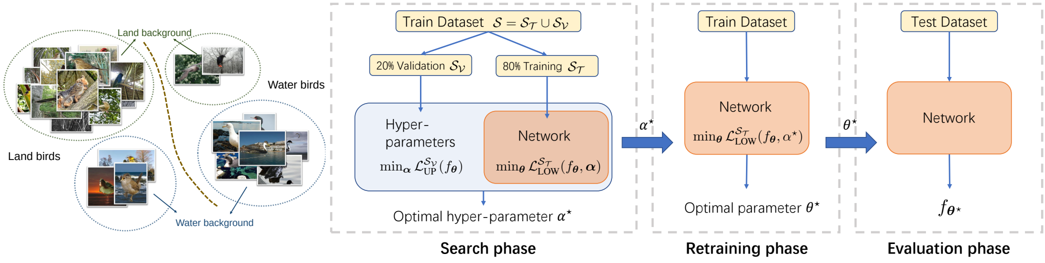
A classical answer to this question is to use a Bayes-consistent training loss function, so that as the sample size grows, the training process returns the Bayes-optimal decision rule. Thus, weighted cross-entropy (e.g., each class gets a different weight, see §4) is traditionally a good choice for optimizing weighted accuracy objectives. Unfortunately, this intuition starts to break down when the training problem is overparameterized, which is a common practice in deep learning: in essence, for large capacity deep nets, the training process can perfectly fit to the data, and training loss is no longer indicative of test error. In fact, recent works [8, 40] show that weighted cross-entropy has minimal benefit to balanced accuracy, and instead alternative methods based on margin adjustment can be effective (namely, by ensuring that minority classes are further away from decision boundary). These ideas led to the development of a parametric cross-entropy function $\ell(y, f(\boldsymbol{x})) = w_y \log\left(1 + \sum_{k \neq y} e^{l_k - l_y} \cdot e^{\Delta_k f_k(\boldsymbol{x}) - \Delta_y f_y(\boldsymbol{x})}\right)$, which allows for a personalized treatment of the individual classes via the design parameters $(w_k, l_k, \Delta_k)_{k=1}^{K}$ [8, 40, 56, 37, 71]. Here, w_k is the classical weighting term whereas l_k and Δ_k are additive and multiplicative logit adjustments. However, despite these developments, it is unclear how such parametric cross-entropy functions can be tuned for use for different fairness objectives, for example to tackle class or group imbalances. The works by [8, 56] provide theoretically-motivated choices for (w_k, l_k), while [40] argues that (w_k, l_k) is not as effective as Δ_k in the interpolating regime of zero training error. However, these works do not provide an optimized loss function that can be systematically tailored for different fairness objectives, such as balanced accuracy common in class imbalanced scenarios, or equal opportunity [24, 18] which is relevant in group-sensitive settings.

In this work, we address these shortcomings by designing the loss function within the optimization *in a principled fashion*, to handle different fairness-seeking objectives. Our main idea is to use bi-level optimization, where the model weights are optimized over the training data, and the loss function is automatically tuned by monitoring the validation loss. Our core intuition is that unlike training data, the validation data is difficult to fit and will provide a consistent estimator of the test objective.

Contributions. Based on this high-level idea, this paper takes a step towards a systematic treatment of imbalanced learning problems with contributions along several fronts: state-of-the-art performance, data augmentation, applications to different imbalance types, and theoretical intuitions. Specifically:

- We introduce *AutoBalance* —a bilevel optimization framework— that designs a fairness-seeking loss function by jointly training the model and the loss function hyperparameters in a systematic way (Figure 1b, Section 4). To further improve the test performance, our design also incorporates *data augmentation policies* personalized to subpopulations (classes or groups). We demonstrate the benefits of AutoBalance when optimizing various fairness-seeking objectives over the state-of-the-art, such as logit-adjustment (LA) [56] and label-distribution-aware margin (LDAM) [8] losses.

- Extensive experiments provide several takeaways (Section 5). First, the impact of individual design parameters in the loss function is revealed, with the additive adjustment l_k and multiplicative adjustment Δ_k synergistically improving the fairness objective. Second, personalized data augmentation can further improve the performance over a single generic augmentation policy. Third, the loss functions designed by AutoBalance are transferable across datasets.


Land background
Water birds
Land birds
Water background
Train Dataset $\mathcal{S}=\mathcal{S}_\mathcal{T}\cup\mathcal{S}_\mathcal{V}$
20% Validation $\mathcal{S}_\mathcal{V}$
80% Training $\mathcal{S}_\mathcal{T}$
Hyper-parameters
$\min_{\boldsymbol{\alpha}} \mathcal{L}_{\text{UP}}^{\mathcal{S}_\mathcal{V}}(f_{\boldsymbol{\theta}})$
Network
$\min_{\boldsymbol{\theta}} \mathcal{L}_{\text{LOW}}^{\mathcal{S}_\mathcal{T}}(f_{\boldsymbol{\theta}},\boldsymbol{\alpha})$
Optimal hyper-parameter $\alpha^\star$
Search phase
$\alpha^\star$
Train Dataset
Network
$\min_{\boldsymbol{\theta}} \mathcal{L}_{\text{LOW}}^{\mathcal{S}_\mathcal{T}}(f_{\boldsymbol{\theta}},\alpha^\star)$
Optimal parameter $\theta^\star$
Retraining phase
$\theta^\star$
Test Dataset
Network
$f_{\boldsymbol{\theta}^\star}$
Evaluation phase

(a) (b)

Figure 1: (a) Example group-imbalance on the Waterbirds dataset [67, 78]. Groups correspond to the distinct background types, while classes are distinct bird types. (b) Framework overview. The search phase conducts a bilevel optimization to design the optimal training loss function parameterized by $\boldsymbol{\alpha}^\star$ by minimizing the validation loss, using a train-validation split (e.g. 80%-20%). The retrain phase uses the original training data and $\boldsymbol{\alpha}^\star$ to obtain the optimal model parameters $\boldsymbol{\theta}^\star$. The evaluation phase predicts the test data using $\boldsymbol{\theta}^\star$.

- Beyond class imbalance, we consider applications of loss function design to the group-sensitive setting (Section 6). Our experiments show that AutoBalance consistently outperforms various baselines, leading to a more efficient Pareto-frontier of accuracy-fairness tradeoffs.

2 Related Work

Our work relates to imbalanced classification, fairness, bilevel optimization, and data augmentation. Below we focus on the former three and defer the extended discussion to the supplementary.

Long-tailed learning. Learning with long-tailed data has received substantial interest historically, with classical methods focusing on designing sampling strategies, such as over- or under-sampling [42, 63, 9, 43, 1, 59, 72, 5, 53]. Several loss re-weighting schemes [57, 55, 28, 13, 6, 36] have been proposed to adjust weights of different classes or samples during training. Another line of work [76, 39, 56] focuses on post-hoc correction. More recently, several works [47, 37, 17, 36, 8, 13, 56, 71] develop more refined class-balanced loss functions (e.g. (4.1)) that better adapt to the training data. In addition, several works [32, 79] point out that separating the representation learning and class balancing can lead to improvements. In this work, our approach is in the vein of class-balanced losses; however, rather than fixing a balanced loss function (e.g. based on the class probabilities in the training dataset), we employ our Algorithm 1 to automatically guide the loss design.

Group-sensitive and Fair Learning. The group-sensitive learning aims to ensure the fairness of the classifier under setups where there exists under-represented groups (e.g., gender, race). [7, 24, 74, 68] propose several fairness metrics as well as insightful methodologies. A line of research [4, 20] optimize the worst-case loss over the test distribution and further applications motivate (label, group) metrics such as equality of opportunity [24, 18] (also recall DEO (3.1)). [61] discusses group-sensitive learning in an over-parameterized regime and proposes that strong regularization ensures fairness. Closer to our work, [61, 40] also study Waterbirds dataset. Compared to the regularization-based approach of [61], we explore a parametric loss design (which is inspired from [40]) to optimize fairness-risk over validation. [18] proposes methods and statistical guarantees for fair empirical risk minimization. A key observation of our work is that, such guarantees based on training-only optimization can be vacuous in the overparameterized regime. Thus, using train-validation split (e.g. our Algo 1) is critical for optimizing fairness metrics more reliably. This is verified by the effectiveness of our approach in the evaluations of Section 6.

Bilevel Optimization. Classical approaches [65] for hyper-parameter optimization are typically based on derivative-free schemes, including random search [69] and reinforcement learning [81, 3, 70, 77]. Recently, a growing line of works focus on differentiable algorithms that are often faster and can scale up to millions of parameters [49, 52, 62, 30, 51]. These techniques [48, 38, 80, 50] with continuous relaxations have shown significant success in neural architecture search, learning rate scheduling, regularization, etc. These methods are typically formulated as a bilevel optimization problem: the upper and lower optimizations minimize the validation and training losses, respectively. Some theoretical guarantees (albeit restrictive) are also available [11, 22, 2, 58]. Different from these, our work focuses on principled design of training loss function to optimize fairness-seeking objectives for imbalanced data. Here, a key algorithmic distinction (e.g. compared to architecture search) is that, our loss function design is only used during optimization and not during inference.

This leads to a more sophisticated hyper-gradient and necessitates additional measures to ensure stability of our approach (see Algo 1).

3 Problem Setup

Let $[K]$ denote the set $\{1, \ldots, K\}$. Let 1_E be the indicator function of event E. Suppose we have a dataset $\mathcal{S} = (\boldsymbol{x}_i, y_i, g_i)_{i=1}^n$ sampled i.i.d. from a distribution $\mathcal{D}$ with input space $\mathcal{X}$, K classes and G groups. For an example $(\boldsymbol{x}, y, g)$, $\boldsymbol{x} \in \mathcal{X}$ is the input features, $y \in [K]$ is the output label, and $g \in [G]$ is the group membership. Let $f : \mathcal{X} \to \mathrm{R}^K$ be a model that outputs a distribution over classes and let $\hat{y}_f(\boldsymbol{x}) = \arg\max_{i \in [K]} f(\boldsymbol{x})$. Standard classification error is defined as $\mathcal{E}(f) = \mathrm{P}_{\mathcal{D}}[y \neq \hat{y}_f(\boldsymbol{x})]$. Let $\ell(y, \hat{y})$ be a differentiable proxy for 0-1 loss (specifically cross-entropy). We similarly denote

$$\text{Population risk: } \mathcal{L}(f) = \mathrm{E}_{\mathcal{D}}[\ell(y, \hat{y}_f(\boldsymbol{x}))], \quad \text{Empirical risk: } \mathcal{L}^{\mathcal{S}}(f) = \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, \hat{y}_f(\boldsymbol{x}_i)).$$

For $(\boldsymbol{x}, y, g) \sim \mathcal{D}$, define the class, group, and (class, group) frequencies as follows

$$\boldsymbol{\pi}_k = \mathrm{P}_{\mathcal{D}}(y = k), \;\; \bar{\boldsymbol{\pi}}_j = \mathrm{P}_{\mathcal{D}}(g = j), \;\; \text{and} \;\; \boldsymbol{\pi}_{k,j} = \mathrm{P}_{\mathcal{D}}(y = k, g = j), \quad \text{for} \quad (k, j) \in [K] \times [G].$$

Similarly, given sample $(\boldsymbol{x}, y, g)$ from $\mathcal{D}$, let $\mathcal{D}_k$ be the distribution conditioned on class $y = k$, let $\bar{\mathcal{D}}_j$ be the distribution conditioned on group $g = j$, let $\mathcal{D}_{k,j}$ be the distribution of $(\boldsymbol{x}, y, g)$ conditioned on $y = k$ and $g = j$. We say that a problem has class (or group) imbalance if majority class (or group) is substantially more frequent than minority. We introduce

$$\text{Class-conditional risk: } \mathcal{L}_k(f) = \mathrm{E}_{\mathcal{D}_k}[\ell(y, \hat{y}_f(\boldsymbol{x}))], \quad \text{Balanced risk: } \mathcal{L}_{\text{bal}}(f) = \frac{1}{K}\sum_{k=1}^{K} \mathcal{L}_k(f).$$

Similarly, group and (class, group)-conditional risks are denoted via $\bar{\mathcal{L}}_j(f)$ and $\mathcal{L}_{k,j}(f)$ respectively. We restrict our attention to the following applications that can benefit from our approach.

✓ **Setting A: Imbalanced classes.** This occurs when class frequencies differ, i.e., $\max_{i \in [K]} \boldsymbol{\pi}_i \gg \min_{i \in [K]} \boldsymbol{\pi}_i$. In this setting, we ignore group membership and focus on classes. Instead of standard accuracy, we will optimize a class-balanced error $\mathcal{E}_{\text{bal}}(f)$ by designing a class-personalized loss.

✓ **Setting B: Imbalanced groups.** This occurs when group or (class, group) frequencies differ, i.e., $\max_{j \in [G]} \bar{\boldsymbol{\pi}}_j \gg \min_{j \in [G]} \bar{\boldsymbol{\pi}}_j$ or $\max_{(k,j)} \boldsymbol{\pi}_{k,j} \gg \min_{(k,j)} \boldsymbol{\pi}_{k,j}$. Specifically, in the fairness literature, groups represent sensitive or protected attributes. A typical goal is ensuring that the prediction of the model is independent of these attributes. While many fairness metrics exist, in this work, we focus on the Difference of Equal Opportunity (DEO) [24, 18]. Our evaluations also focus on binary classification (with labels denoted via ±) and two groups ($K = G = 2$). With this setup, the DEO risk is defined as $\mathcal{L}_{\text{DEO}}(f) = |\mathcal{L}_{+,1}(f) - \mathcal{L}_{+,2}(f)|$. When both classes are equally relevant (rather than $y = +1$ implying a semantically positive outcome), we use the following definition:

$$\mathcal{L}_{\text{DEO}}(f) = |\mathcal{L}_{+,1}(f) - \mathcal{L}_{+,2}(f)| + |\mathcal{L}_{-,1}(f) - \mathcal{L}_{-,2}(f)|. \tag{3.1}$$

4 Loss Function Design and Proposed Method

Our main goal in this paper is automatically designing loss functions to optimize target objectives for imbalanced learning (e.g., Settings A and B). We will employ a parametrizable family of loss functions that can be tailored to the needs of different classes or groups. Cross-entropy variations have been proposed by [47, 37, 17] to optimize balanced objectives. Our design space will utilize recent works by [40, 71], who introduce Vector Scaling (VS)-loss and Class-Dependent Temperatures (CDT). Specifically, for Setting A[1] we build on the following loss function parametrized by three vectors $\boldsymbol{w}, \boldsymbol{l}, \boldsymbol{\Delta} \in \mathrm{R}^K$:

$$\ell(y, f(\boldsymbol{x})) = w_y \log\left(1 + \sum_{k \neq y} e^{l_k - l_y} \cdot e^{\Delta_k f_k(\boldsymbol{x}) - \Delta_y f_y(\boldsymbol{x})}\right). \tag{4.1}$$

This loss is same as VS-loss of [40], which also contains $\boldsymbol{w}, \boldsymbol{l}, \boldsymbol{\Delta} \in \mathrm{R}^K$. However, the denominator of VS-loss uses Δ_y whereas we use Δ_k similar to CDT of [71]. Here, w_y enables conventional weighted CE and l_y is the additive adjustment to the logits. Recently [56] proposed a Fisher consistent logit adjustment (LA) loss parameterized by w_y and l_y; we make the following related observation.

---

[1] Discussion of Setting B (group-imbalance) is deferred to Section 6, however the main ideas are similar.

Algorithm 1: Bilevel Optimization for AutoBalance

```
Input: Model $f_{\boldsymbol{\theta}}$ with weights $\boldsymbol{\theta}$, dataset $\mathcal{S} = \mathcal{S}_{\mathcal{T}} \cup \mathcal{S}_{\mathcal{V}}$, step sizes $\eta_{\boldsymbol{\alpha}}$ & $\eta_{\boldsymbol{\theta}}$, # iterations $t_2 > t_1$
Initialize $\boldsymbol{\alpha}$ with $\ell_{\text{UP}}(\cdot) = \ell_{\text{LOW}}(\cdot;\boldsymbol{\alpha})$        // Consistent initialization
Train $\boldsymbol{\theta}$ for $t_1$ iterations ($\boldsymbol{\alpha}$ is fixed)          // Warm-up training
for $i \leftarrow t_1$ to $t_2$ do
    Sample training batch $\mathcal{B}_{\mathcal{T}}$ from $\mathcal{S}_{\mathcal{T}}$;
    // Apply class-personalized augmentation $\mathcal{B}_{\mathcal{T}} \leftarrow \mathcal{A}(\mathcal{B}_{\mathcal{T}})$ (implicitly)
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta_{\boldsymbol{\theta}} \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\text{LOW}}^{\mathcal{B}_{\mathcal{T}}}(f_{\boldsymbol{\theta}};\boldsymbol{\alpha})$
    Sample validation batch $\mathcal{B}_{\mathcal{V}}$ from $\mathcal{S}_{\mathcal{V}}$;
    Compute hyper-gradient $\nabla_{\boldsymbol{\alpha}} \mathcal{L}_{\text{UP}}^{\mathcal{B}_{\mathcal{V}}}(f_{\boldsymbol{\theta}})$   //via Implicit Differentiation e.g. [49]
    $\boldsymbol{\alpha} \leftarrow \boldsymbol{\alpha} - \eta_{\boldsymbol{\alpha}} \nabla \mathcal{L}_{\text{UP}}^{\mathcal{B}_{\mathcal{V}}}(f_{\boldsymbol{\theta}})$              //Update loss function hyper-parameters
end
Set $\boldsymbol{\alpha}_\star \leftarrow \boldsymbol{\alpha}$, $\mathcal{S}_{\mathcal{T}} \leftarrow \mathcal{S}$, reset weights $\boldsymbol{\theta}$
Train $\boldsymbol{\theta}$ for $t_2$ iterations using $\boldsymbol{\alpha}_\star$          // Final training over full $\mathcal{S}$ using $\boldsymbol{\alpha}_\star$
Result: The final model $\boldsymbol{\theta}_\star \leftarrow \boldsymbol{\theta}$ and hyper-parameters $\boldsymbol{\alpha}_\star$
```

Lemma 1 *Parametric loss function* (4.1) *is not consistent for standard or balanced errors if there are distinct multiplicative adjustments i.e.* $\Delta_i \neq \Delta_j$ *for some* $i, j \in [K]$.

While consistency is a desirable property, it is intuitively more critical during the earlier phase of the training where the training risk is more indicative of the test risk. In the interpolating regime of zero-training error, [40] shows that $\boldsymbol{w}, \boldsymbol{l}$ can be ineffective and multiplicative $\boldsymbol{\Delta}$-adjustment can be more favorable. Our algorithm will be initialized with a consistent weighted-CE; however, we will allow the algorithm to automatically adapt to the interpolating regime by tuning $\boldsymbol{l}$ and $\boldsymbol{\Delta}$.

Proposed training loss function. For our algorithm, we will augment (4.1) with *data augmentation* that can be *personalized to distinct classes*. Let us denote the data augmentation policies by $\mathcal{A} = (\mathcal{A}_y)_{y=1}^K$ where each $\mathcal{A}_y$ stochastically augments an input example with label y. Additionally, we clamp $\boldsymbol{\Delta}_i$ with the sigmoid function σ to limit its range to (0,1) to ensure non-negativity. To this end, our loss function for the lower-level optimization (over training data) is as follows:

$$\ell_{\text{LOW}}(y, \boldsymbol{x}, f; \boldsymbol{\alpha}) = -\,\mathrm{E}_{\mathcal{A}}\left[w_y \log\left(\frac{e^{\sigma(\Delta_y) f_y(\mathcal{A}_y(\boldsymbol{x})) + l_y}}{\sum_{i\in[K]} e^{\sigma(\Delta_i) f_i(\mathcal{A}_y(\boldsymbol{x})) + l_i}} \right)\right]. \tag{4.2}$$

Here, $\boldsymbol{\alpha}$ is the set of hyperparameters of the loss function that we wish to optimize, specifically $\boldsymbol{\alpha} = [\boldsymbol{w}, \boldsymbol{l}, \boldsymbol{\Delta}, \text{param}(\mathcal{A})]$. param$(\mathcal{A})$ is the parameterization of the augmentation policies $(\mathcal{A}_y)_{y\in[K]}$.

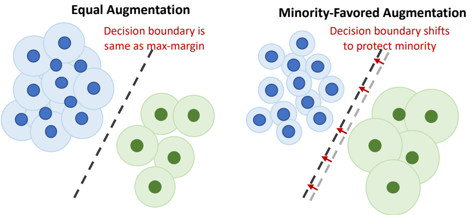
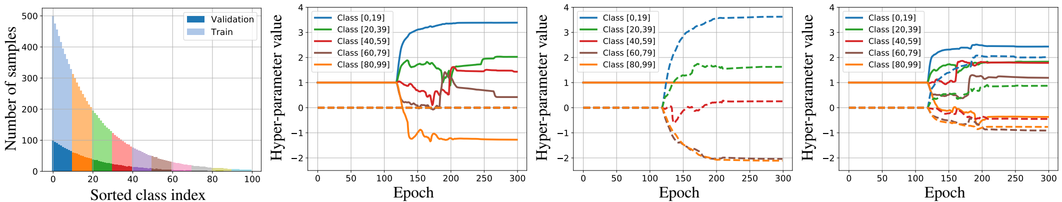

Equal Augmentation
Decision boundary is same as max-margin
Minority-Favored Augmentation
Decision boundary shifts to protect minority

Figure 2: Data augmentation can shift the decision boundary to benefit the minority class by providing a larger margin. Lemma 2 establishes an equivalence between spherical data augmentation and parametric cross-entropy loss.

Personalized data augmentation (PDA). Remarkable benefits of data augmentation techniques provide a natural motivation to investigate whether one can benefit from learning class-personalized augmentation policies. To explain our intuition, consider a spherical augmentation strategy where $\mathcal{A}_y(\boldsymbol{x})$ samples a vector uniformly from an ℓ_2-ball of radius ε_y around $\boldsymbol{x}$. As visualized in Figure 2 for a linear classifier, if the augmentation strengths of both classes are equal, the max-margin classifier is not affected by the application of the data augmentation and remains identical. Thus, augmentation has no benefit. However by applying a stronger augmentation on minority, the decision boundary is shifted to protect minority which can provably benefit the balanced accuracy [40, 8]. The following intuitive observation links the PDA to parametric loss (4.1).

Lemma 2 *Consider a binary classification task with labels* 0 *and* 1 *and a linearly separable training dataset. For any parametric loss* (4.1) *choices of* $(l_i, \Delta_i, w_i)_{i=0}^1$, *there exists spherical augmentation strengths for minority/majority classes so that, without regularization,* ***optimizing the logistic loss with personalized augmentations returns the same classifier as optimizing*** (4.1)**.**

This Lemma is similar in flavor to the result of [33], which considers a larger uncertainty set around the minority class. As discussed in the supplementary materials, Lemma 2 is relevant for



(a) Train-validation sizes for CIFAR-100-LT clusters (b) Evolution of $\boldsymbol{\Delta}$ when only training $\boldsymbol{\Delta}$ (c) Evolution of $\boldsymbol{l}$ when only training $\boldsymbol{l}$ (d) Evolution of $\boldsymbol{\Delta}$ and $\boldsymbol{l}$ when optimizing jointly

Figure 3: (a) Visualizing class clustering and train-validation split. (b), (c), (d) Evolution of loss function parameters $\boldsymbol{l}, \boldsymbol{\Delta}$ over epochs for CIFAR-100-LT where solid curves and dashed curves corresponds to $\boldsymbol{\Delta}$ and $\boldsymbol{l}$ respectively. We display average value of 20 classes for better visualization. Based on theory, the minority classes should be assigned a larger margin. During the initial 120 epochs, we use weighted cross-entropy training and AutoBalance kicks in after epoch 120. Observe that, AutoBalance does indeed learn larger parameters (l_y, Δ_y) for minority class clusters (each containing 20 classes) consistent with theoretical intuition. In all Figures (b), (c), (d), by the end of training, the colors are ordered according to the class frequency. However, when Δ_y is trained jointly with l_y (Fig (d)), the training is more stable compared to training Δ_y alone (Fig (b)). Thus, besides its accuracy benefits in Table 1, l_y also seems to have optimization benefits.

the overparameterized regime whereas the approach of [33] is ineffective for separable data [54]. Algorithmically, the augmentations that we consider are much more flexible than the ℓ_p-balls of [33] and our experiments showcase the value of our approach in state-of-the-art multiclass settings. Finally, we note that, the (theoretical) benefits of PDA can go well-beyond Lemma 2 by leveraging the invariances [10, 14] (via rotation, translation).

4.1 Proposed Bilevel Optimization Method

We formulate the loss function design as a bilevel optimization over $\boldsymbol{\alpha}$ and a hypothesis set $\mathcal{F}$. Split the dataset $\mathcal{S}$ into training $\mathcal{S}_{\mathcal{T}}$ and validation $\mathcal{S}_{\mathcal{V}}$ sets with $n_{\mathcal{T}}$ and $n_{\mathcal{V}}$ examples respectively. The upper-level variable $\boldsymbol{\alpha}$ aims to minimize a desired fairness-seeking objective $\ell_{\text{UP}}(y, \hat{y})$, and the lower-level hypothesis $f \in \mathcal{F}$ aims to minimize the training loss (4.2) as follows:

$$\min_{\boldsymbol{\alpha}} \mathcal{L}_{\text{UP}}^{\mathcal{S}_{\mathcal{V}}}(f_{\boldsymbol{\alpha}}) \quad \text{WHERE} \quad f_{\boldsymbol{\alpha}} = \arg\min_{f \in \mathcal{F}} \mathcal{L}_{\text{LOW}}^{\mathcal{S}_{\mathcal{T}}}(f; \boldsymbol{\alpha}) := \frac{1}{n_{\mathcal{T}}} \sum_{i=1}^{n_{\mathcal{T}}} \ell_{\text{LOW}}(y_i, \boldsymbol{x}_i, f; \boldsymbol{\alpha}). \tag{4.3}$$

Here, $\mathcal{L}_{\text{UP}}$ is obtained as the empirical average of ℓ_{UP}. The function ℓ_{UP} is weighted cross-entropy, chosen to be a consistent proxy for the desired accuracy objective $\mathcal{E}_{\text{UP}}$. For instance, if $\mathcal{E}_{\text{UP}}$ is a superposition of standard and balanced errors $\mathcal{E}_{\text{UP}} = (1-\lambda)\mathcal{E} + \lambda\mathcal{E}_{\text{bal}}$, then similarly $\mathcal{L}_{\text{UP}} = (1-\lambda)\mathcal{L} + \lambda\mathcal{L}_{\text{bal}}$. Algorithm 1 summarizes our approach and highlights the key components. The training loss $\ell_{\text{LOW}}(\cdot; \boldsymbol{\alpha})$ is also initialized as a consistent proxy for $\mathcal{E}_{\text{UP}}$ (e.g. same as ℓ_{UP}).

Implicit Differentiation and Warm-up training. For a loss function parameter $\boldsymbol{\alpha}$, the hyper-gradient can be written via the chain-rule $\frac{\partial \mathcal{L}_{\text{UP}}(\boldsymbol{\theta}^\star)}{\partial \boldsymbol{\alpha}} = \frac{\partial \mathcal{L}_{\text{UP}}(\boldsymbol{\theta}^\star)}{\partial \boldsymbol{\theta}^\star} \frac{\partial \boldsymbol{\theta}^\star}{\partial \boldsymbol{\alpha}}$ Here, $\boldsymbol{\theta}^\star$ is the solution of the lower-level problem. We note that $\partial \mathcal{L}_{\text{UP}}/\partial \boldsymbol{\alpha} = 0$ since $\boldsymbol{\alpha}$ does not appear within the upper-level loss. Also observe that $\partial \mathcal{L}_{\text{UP}}(\boldsymbol{\theta}^\star)/\partial \boldsymbol{\theta}^\star$ can be directly computed by taking the gradient. To compute $\partial \boldsymbol{\theta}^\star/\partial \boldsymbol{\alpha}$, we follow the recent work by [49] and employ the Implicit Function Theorem (IFT). If there exists a fixed point $(\boldsymbol{\theta}^\star, \boldsymbol{\alpha}^\star)$ that satisfies $\partial \mathcal{L}_{\text{LOW}}(\boldsymbol{\theta}^\star, \boldsymbol{\alpha}^\star)/\partial \boldsymbol{\theta} = 0$ and regularity conditions are satisfied, then around $\boldsymbol{\alpha}^\star$, there exists a function $\boldsymbol{\theta}(\boldsymbol{\alpha})$ such that $\boldsymbol{\theta}(\boldsymbol{\alpha}^\star) = \boldsymbol{\theta}^\star$ and we also have $\frac{\partial \boldsymbol{\theta}}{\partial \boldsymbol{\alpha}} = (\frac{\partial^2 \mathcal{L}_{\text{LOW}}}{\partial \boldsymbol{\theta}^2})^{-1} \frac{\partial^2 \mathcal{L}_{\text{LOW}}}{\partial \boldsymbol{\theta} \partial \boldsymbol{\alpha}}$. However, directly computing inverse Hessian $(\frac{\partial^2 \mathcal{L}_{\text{LOW}}}{\partial \boldsymbol{\theta}^2})^{-1}$ is usually time consuming or even impossible for modern neural networks which have millions of parameters. To compute the hyper-gradient while avoiding extensive computation, we approximate the inverse Hessian via the Neumann series, which is widely used for inverse Hessian estimation [45, 49]. Finally, the warm-up phase of our method (Line 2 of Algo. 1) is essential to guarantee that the IFT assumption $\frac{\partial \mathcal{L}_{\text{LOW}}(\boldsymbol{\theta}^\star, \boldsymbol{\alpha}^\star)}{\partial \boldsymbol{\theta}} = 0$ is approximately satisfied.

Why is train-validation split critical? It is well-understood that large capacity neural networks can perfectly fit and achieve 100% training accuracy [19, 82, 31]. This also implies that over the training data, different accuracy metrics or fairness constraints can be perfectly satisfied (e.g. 100% balanced accuracy, and 0% DEO). To truly find a model that lies on the Pareto-front of the (accuracy,

Method	CIFAR-10-LT	CIFAR100-LT	ImageNet-LT	iNaturalist
Cross-Entropy	30.33	62.69	60.81	39.76
LDAM loss [8]	26.45	59.40	58.14	35.42
LA loss ($\tau = 1$) [56]	23.32	58.92	55.60	34.37
Algo. 1: $\boldsymbol{\alpha} \leftarrow \tau$ of LA loss	21.75	58.76	55.16	34.18
Algo. 1: $\boldsymbol{\alpha} \leftarrow \boldsymbol{l}$	22.86	58.73	55.39	34.42
Algo. 1: $\boldsymbol{\alpha} \leftarrow \boldsymbol{\Delta}$	22.52	58.49	54.72	34.14
Algo. 1: $\boldsymbol{\alpha} \leftarrow \boldsymbol{\Delta}\&\boldsymbol{l}$	21.39	56.77	52.90	33.16
Algo. 1: $\boldsymbol{\alpha} \leftarrow \boldsymbol{\Delta}\&\boldsymbol{l}$, LA init	**21.16**	**56.71**	**51.96**	**33.09**

Table 1: Evaluations of balanced accuracy on long-tailed data. Algo. 1 with $\boldsymbol{\Delta}\&\boldsymbol{l}$ design space and LA initialization (bottom row) outperforms other baselines, across various datasets.

fairness) tradeoff, the optimization procedure should (approximately) evaluate on the population loss. To this end, the validation phase provides this crucial population loss-proxy in the overparameterized setting where training error can be vacuous. Following the literature [34, 35], the intuition is that, as the dimensionality of the hyper-parameter $\boldsymbol{\alpha}$ is typically smaller than the validation size $n_{\mathcal{V}}$, validation loss will not overfit and will be indicative of the test even if the training loss is zero. In the supplementary materials, we formalize this intuition with emphasis on multi-objective tradeoffs; namely, under intuitive stability conditions, we show that a small amount of validation data is sufficient to ensure that the Pareto-front of the validation risk uniformly approximates that of the test-risk. Concretely, for two objectives $(\mathcal{L}_1, \mathcal{L}_2)$, uniformly over all λ, the $\boldsymbol{\alpha}$ minimizing the validation risk $\mathcal{L}_{\text{UP}}^{\mathcal{S}_\mathcal{V}}(f) = (1-\lambda)\mathcal{L}_1^{\mathcal{S}_\mathcal{V}} + \lambda\mathcal{L}_2^{\mathcal{S}_\mathcal{V}}$ in (4.3) also approximately minimizes the test risk $\mathcal{L}_{\text{UP}}(f)$.

5 Evaluations for Imbalanced Classes

In this section, we present our experiments on various datasets (CIFAR-10, CIFAR-100, iNaturalist-2018 and ImageNet) when the classes are imbalanced. The goal is to understand whether our bilevel optimization can design effective loss functions that improve balanced error $\mathcal{E}_{\text{bal}}$ on the test set. The setup is as follows. $\mathcal{E}_{\text{bal}}$ is the test objective. The validation loss $\mathcal{L}_{\text{UP}}$ is the balanced cross-entropy CE_{bal}. We consider various designs for ℓ_{LOW} such as individually tuning $\boldsymbol{w}, \boldsymbol{l}, \boldsymbol{\Delta}$ and augmentation. We report the average result of 3 random experiments under this setup.

Datasets. We follow previous works [56, 13, 8] to construct long-tailed versions of the datasets. Specifically, for a K-class dataset, we create a long-tailed dataset by reducing the number of examples per class according to the exponential function $n_i' = n_i\mu^i$, where n_i is the original number of examples for class i, n_i' is the new number of examples per class, and $\mu < 1$ is a scaling factor. Then, we define the imbalance factor $\rho = n_0'/n_K'$, which is the ratio of the number of examples in the largest class (n_0') to the smallest class (n_K'). For the **CIFAR-10-LT** and **CIFAR-100-LT** dataset, we construct long-tailed versions of the datasets with imbalance factor $\rho = 100$. **ImageNet-LT** contains 115,846 training examples and 1,000 classes, with imbalance factor $\rho = 256$. **iNaturalist-2018** contains 435,713 images from 8,142 classes, and the imbalance factor is $\rho = 500$. These choices follow that of [56]. For all datasets, we split the long-tailed training set into 80% training and 20% validation during the search phase (Figure 1b).

Implementation. In both CIFAR datasets, the lower-level optimization trains a ResNet-32 model with standard mini-batch stochastic gradient decent (SGD) using learning rate 0.1, momentum 0.9, and weight decay $1e-4$, over 300 epochs. The learning rate decays at epochs 220 and 260 with a factor 0.1. The upper-level hyper-parameter optimization computes the hyper-gradients via implicit differentiation. Because the hyper-gradient is mostly meaningful when the network achieves near zero loss (Thm 1 of [49]), we start the upper optimization after 120 epochs of the lower-level optimization, using SGD with initial learning rate 0.01, momentum 0.9, and weight decay $1e-4$. For CIFAR-LT, 20 hyper-parameters are trained, corresponding to $\boldsymbol{l}$ and $\boldsymbol{\Delta}$ for all classes. For CIFAR-100, because the size of validation data in minority classes can be too small (e.g., only one example in a tail class), as visualized in Figure 3(a), we gather classes of similar frequencies into clusters of size 10 to share the same hyper-parameters (i.e., the values and updates of (l_y, Δ_y)'s). Similarly, for ImageNet-LT, we gather classes into clusters of size 10, and for iNaturalist, we gather classes into clusters of size 40. For ImageNet-LT and iNaturalist, following previous work [56], we use ResNet-50 and SGD for the lower and upper optimizations, and the same learning rate as CIFAR-LT over 150 epochs. We

Method	CIFAR-10-LT	CIFAR100-LT	ImageNet-LT
MADAO [26]	24.39	59.10	59.92
Algo. 1: $\boldsymbol{\alpha} \leftarrow$PDA	22.53	58.55	58.77
Algo. 1: $\boldsymbol{\alpha} \leftarrow$PDA, $\boldsymbol{\Delta}\&\boldsymbol{l}$	20.76	56.49	52.11

Table 2: The result of personalized data optimization.

Source\Target	CIFAR-10-LT	CIFAR100-LT	ImageNet-LT
CIFAR-10-LT	21.39	58.12	56.13
CIFAR100-LT	22.32	56.77	55.89
ImageNet-LT	23.85	59.17	52.90

Table 3: The results of hyper-parameter transfer.

train the network for 40 epochs to warm up before the loss function design starts. The learning rate decays by a factor 0.1 at epochs 80 and 120.

Personalized Data Augmentation (PDA). For PDA, we utilize the AutoAugment [12] policy space and apply a bilevel search for the augmentation policy. Our approach follows existing differentiable augmentation strategies (e.g, [26]); however, we train separate policies for each class cluster to ensure that the resulting policies can adjust to class frequencies. Due to space limitations, please see supplementary materials for further details.

Results and discussion. We compared our methods with the state-of-the-art long-tail learning methods. Table 1 shows the results of our experiments where the design space is parametric CE (4.1). In the first part of the table, we conduct experiments for three baseline methods: normal CE, LDAM [8] and Logit Adjustment loss with temperature parameter $\tau = 1$ [56]. The latter choice guarantees Fisher consistency. In the second part of the Table 1, we study Algo. 1 with design spaces $\boldsymbol{l}$, $\boldsymbol{\Delta}$, and $\boldsymbol{l}\&\boldsymbol{\Delta}$. The first version of Algo. 1 in Table 1 tunes the LA loss parameter τ where $\boldsymbol{l}$ is parameterized by a single scalar τ as $l_y = \tau \log(\pi_y)$. The next three versions of Algo. 1 consider tuning $\boldsymbol{l}$, $\boldsymbol{\Delta}$, $\boldsymbol{l}\&\boldsymbol{\Delta}$ respectively (Figure 3b-d shows the evolution of the $\boldsymbol{l}$ and $\boldsymbol{\Delta}$ parameters during the optimization). Finally, in last version of Algo. 1, the loss design is initialized with LA loss with $\tau = 1$ (rather than balanced CE). The takeaway from these results is that our approach consistently leads to a superior balanced accuracy objective. That said, tuning the LA loss alone is highly competitive with optimizing $\boldsymbol{\Delta}$ and $\boldsymbol{l}$ alone (in fact, strictly better for CIFAR-10-LT, indicating Algo. 1 does not always converge to the optimal design). Importantly, when combining $\boldsymbol{l}\&\boldsymbol{\Delta}$, our algorithm is able to design a better loss function and outperform all rows across all benchmarks. Finally, when the algorithm further is initialized with LA loss, the performance further improves accuracy, demonstrating that warm-starting with good designs improves performance.

In Table 2, we study the benefits of data augmentation, following our intuitions from Lemma 2. We compare to the differentiable augmentation baseline of MADAO [26] which trains a single policy for the full dataset. PDA is a personalized variation of MADAO and leads to noticeable improvement across all benchmarks (most noticeably in CIFAR-10-LT). More importantly, the last line of the table demonstrates that PDA can be synergistically combined with the parametric CE (4.1) which leads to further improvements. Finally, in Table 3, we investigate the transferability of our loss design (including augmentation). Here, a class within the new dataset uses the loss function designed for the closest class from the old dataset, where closest means the most similar class frequency in terms of percentile (when all classes of both datasets are sorted). These results demonstrate that AutoBalance designs transferable loss functions; e.g., loss functions transferred from CIFAR-LT to ImageNet have only slight performance degradation, compared to training the loss from scratch (the diagonals).

6 Approaches and Evaluations for Imbalanced Groups

While Section 5 focuses on the fundamental challenge of balanced error minimization, a more ambitious goal is optimizing generic fairness-seeking objectives. In this section, we study accuracy-fairness tradeoffs by examining the pareto-frontiers of the DEO (3.1), group-balanced error, and standard error. We also investigate group-balanced risk defined as $\mathcal{L}^{\mathcal{G}}_{\text{bal}}(f) = \frac{1}{KG}\sum_{k=1}^{K}\sum_{j=1}^{G}\mathcal{L}_{k,j}(f)$. Note that this definition treats each (class, group) pair as its own (sub)group. Throughout, we explicitly set the loss to cross-entropy for clarity, thus we use CE, $\text{CE}^{\mathcal{G}}_{\text{bal}}$, CE_{DEO} to refer to $\mathcal{L}$, $\mathcal{L}^{\mathcal{G}}_{\text{bal}}$, $\mathcal{L}_{\text{DEO}}$.

Validation (upper-level) loss function. In Algo. 1, we set $\mathcal{L}_{\text{UP}} = (1-\lambda)\cdot\text{CE} + \lambda\cdot\text{CE}_{\text{DEO}}$ for varying $0 \le \lambda \le 1$. The parameter λ enables a trade-off between accuracy and fairness.

Group-sensitive training loss design. The parametric cross-entropy (CE) can be extended to (class, group) imbalance by extending hyper-parameter $\boldsymbol{\alpha}$ to $[K]\times[G]$ variables $\boldsymbol{w}, \boldsymbol{l}, \boldsymbol{\Delta} \in \mathrm{R}^{[K]\times[G]}$ generalizing (4.1), (4.2). This leads us to the following design:

$$\ell_{low}(y, g, f(\boldsymbol{x}); \boldsymbol{\alpha}) = -w_{yg}\log\left(\frac{e^{\sigma(\Delta_{yg})f_y(\boldsymbol{x})+\iota_{yg}}}{\sum_{k\in[K]} e^{\sigma(\Delta_{kg})f_k(\boldsymbol{x})+\iota_{kg}}}\right). \tag{6.1}$$

Loss function	Balanced Error	Worst (class, group) error	DEO
Cross entropy (CE)	23.38	43.25	33.75
CE_{bal}	20.83	36.67	20.25
Group-LA loss	22.83	40.50	29.33
CE_{DEO}	19.29	35.17	25.25
$0.1 \cdot \text{CE} + 0.9 \cdot \text{CE}_{\text{DEO}}$ ($\lambda = 0.1$)	20.06	31.67	26.25
Algo. 1: with $\lambda = 0.1$	**15.13**	**30.33**	**4.25**

Table 4: Comparison of fairness metrics for group-imbalanced experiments. The first five rows are different training loss choices, where CE_{bal}, Group-LA, and CE_{DEO} promote group fairness. The last row is Algo. 1, which designs training loss for the validation loss choice of $0.1 \cdot \text{CE} + 0.9 \cdot \text{CE}_{\text{DEO}}$. We note that, DEO can be trivially minimized by always predicting the same class. To avoid this, we use a mild amount of CE loss with $\lambda = 0.1$ in Algo. 1.

Here, w_{yg} applies weighted CE, while Δ_{yg} and ι_{yg} are logit adjustments for different (class, groups). Note that a similar group-sensitive loss is proposed in [40] for binary classification.

Baselines. We will compare Algo. 1 with training loss functions parameterized via $(1-\lambda)\cdot\text{CE}+\lambda\cdot\mathcal{L}_{\text{reg}}$. Here $\mathcal{L}_{\text{reg}}$ is a fairness-promoting regularization. Specifically, as displayed in Table 4 and Figure 4, we will set $\mathcal{L}_{\text{reg}}$ to be $\text{CE}^{\mathcal{G}}_{\text{bal}}$, CE_{DEO} and Group LA. "Group LA" is a natural generalization of the LA loss to group-sensitive setting; it chooses weights $w_g = 1/\bar{\pi}_g$ to balance group frequencies and then applies logit-adjustment with $\tau = 1$ over the classes conditioned on the group-membership: $\ell(y, g, f(\boldsymbol{x})) = -\frac{1}{G\bar{\boldsymbol{\pi}}_g} \log \left(\frac{e^{f_y(\boldsymbol{x})+\tau \log \pi_{yg}}}{\sum_{k\in[K]} e^{f_k(\boldsymbol{x})+\tau \log \pi_{kg}}} \right)$.

Datasets. We experiment with the modified Waterbird dataset [61]. The goal is to correctly classify the bird type despite the spurious correlations due to the image background. The distribution of the original data is as follows. The binary classes $k \in \{-, +\}$ correspond to $\{\text{waterbird}, \text{landbird}\}$, and the groups $[G] = \{1, 2\}$ correspond to $\{\text{land background}, \text{water background}\}$. The fraction of data in each (class, group) pair is $\boldsymbol{\pi}_{-,2} = 0.22$, $\boldsymbol{\pi}_{-,1} = 0.012$, $\boldsymbol{\pi}_{+,2} = 0.038$, and $\boldsymbol{\pi}_{+,1} = 0.73$. The landbird on the water background ($\{+, 2\}$) and the waterbird on the land background ($\{-, 1\}$) are minority sub-groups within their respective classes. The test set, following [61], has equally allocated bird types on different backgrounds, i.e., $\boldsymbol{\pi}_{\pm,j} = 0.25$. As the test dataset is balanced, the standard classification error $\mathcal{E}(f)$ is defined to be the weighted error $\mathcal{E}(f) = \boldsymbol{\pi}_{y,g}\mathcal{E}_{\text{y,g}}(f)$.

Implementation. We follow the feature extraction method from [61], where x_i are 512-dimensional ResNet18 features. When using Algo. 1, we split the original training data into 50% training and 50% validation. The search phase uses 150 epochs of warm up followed by 350 epochs of bilevel optimization. The remaining implementation details are similar to Section 5.

Results and discussion. We consider various fairness-related metrics, including the worst (class, group) error, DEO $\mathcal{E}_{\text{DEO}}(f)$ and the balanced error $\mathcal{E}^{\mathcal{G}}_{\text{bal}}(f)$.We seek to understand whether AutoBalance algorithm can improve performance on the test set compared to the baseline training loss functions of the form $(1-\lambda)\cdot\text{CE}+\lambda\cdot\mathcal{L}_{\text{reg}}$. In Figure 4, we show the influence of the parameter λ where $\mathcal{L}_{\text{reg}}$ is chosen to be CE_{DEO}, CE_{bal}, or Group-LA (each point on the plot represents a different λ value). As we sweep across values of λ, there arises a tradeoff between standard classification error $\mathcal{E}(f)$ and the fairness metrics. We observe that Algo. 1 significantly Pareto-dominates alternative approaches, for example achieving lower DEO or balanced error for the same standard error. This demonstrates the value of automatic loss function design for a rich class of fairness-seeking objectives.

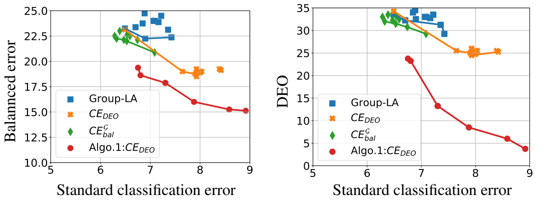


Figure 4: Waterbirds fairness-accuracy tradeoffs for parametrized loss designs $(1-\lambda)\cdot\text{CE}+\lambda\cdot\mathcal{L}_{\text{reg}}$, for different $\mathcal{L}_{\text{reg}}$ choices. Group-balanced error $\mathcal{E}^{\mathcal{G}}(f)$ (left) and DEO $\mathcal{E}_{\text{DEO}}(f)$ (right) are plotted as a function of the misclassification error $\mathcal{E}(f)$. Algo. 1 exhibits a noticeably better tradeoff curve as it uses a DEO-based validation objective to design an optimized training loss function.

Next, in Table 4, we solely focus on optimizing the fairness objectives (rather than standard error). Thus, we simply set $\lambda = 0$ and compare against CE_{DEO}, CE_{bal}, Group-LA as the baseline approaches (without blending CE). We also display the outcome of Algo. 1 with $\lambda = 0.1$. Similar too Figure 4, our approach outperforms all baselines for all metrics. The performance gap is particularly noticeable when it comes to DEO (4.25% for Algo. 1, vs 20.25% for the best baseline).

7 Conclusions and Future Directions

This work provides an optimization-based approach to automatically design loss functions to address imbalanced learning problems. Our algorithm consistently outperforms, or is at least competitive with, the state-of-the-art approaches for optimizing balanced accuracy. Importantly, our approach is not restricted to imbalanced classes or specific objectives, and can achieve good tradeoffs between (accuracy, fairness) on the Pareto frontier. We also provide theoretical insights on certain algorithmic aspects including loss function design, data augmentation, and train-validation split.

Potential Limitations, Negative Societal Impacts, & Precautions: Our algorithmic approach can be considered within the realm of automated machine learning literature (AutoML) [29]. AutoML algorithms often optimize the model performance, thus reducing the need for engineering expertise at the expense of increased computational cost and increased carbon footprint. For instance, our procedure is computationally more intensive compared to the theory-inspired loss function prescriptions of [56, 8]. A related limitation is that Algo. 1 can be brittle in extremely imbalanced scenarios with very few samples per class. We took the several steps to help mitigate such issues: first, our algorithm is initialized with a Bayes consistent loss function to provide a warm-start (such as the proposal of [56]). Second, we reduce the hyper-parameter search space by grouping classes with similar frequencies (to help with brittleness). Finally, evaluations show that the designed loss functions are transferable, and hence our algorithm does does not have to train from scratch for a new dataset.

References

[1] Shin Ando and Chun Yuan Huang. Deep over-sampling framework for classifying imbalanced data. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 770–785. Springer, 2017.

[2] Yu Bai, Minshuo Chen, Pan Zhou, Tuo Zhao, Jason D Lee, Sham Kakade, Huan Wang, and Caiming Xiong. How important is the train-validation split in meta-learning? *arXiv preprint arXiv:2010.05843*, 2020.

[3] Bowen Baker, Otkrist Gupta, Nikhil Naik, and Ramesh Raskar. Designing neural network architectures using reinforcement learning. *arXiv preprint arXiv:1611.02167*, 2016.

[4] Aharon Ben-Tal, Dick den Hertog, Anja De Waegenaere, Bertrand Melenberg, and Gijs Rennen. Robust solutions of optimization problems affected by uncertain probabilities. *Management Science*, 59(2):341–357, 2013.

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

[6] Jonathon Byrd and Zachary Lipton. What is the effect of importance weighting in deep learning? In *International Conference on Machine Learning*, pages 872–881. PMLR, 2019.

[7] Toon Calders, Faisal Kamiran, and Mykola Pechenizkiy. Building classifiers with independency constraints. In *2009 IEEE International Conference on Data Mining Workshops*, pages 13–18. IEEE, 2009.

[8] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. *arXiv preprint arXiv:1906.07413*, 2019.

[9] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[10] Shuxiao Chen, Edgar Dobriban, and Jane H Lee. A group-theoretic framework for data augmentation. *Journal of Machine Learning Research*, 21(245):1–71, 2020.

[11] Nicolas Couellan and Wenjuan Wang. On the convergence of stochastic bi-level gradient methods. *Optimization*, 2016.

[12] ED Cubuk, B Zoph, D Mane, V Vasudevan, and QV Le. Autoaugment: Learning augmentation policies from data. arxiv 2018. *arXiv preprint arXiv:1805.09501*.

[13] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9268–9277, 2019.

[14] Tri Dao, Albert Gu, Alexander Ratner, Virginia Smith, Chris De Sa, and Christopher Ré. A kernel theory of modern data augmentation. In *International Conference on Machine Learning*, pages 1528–1537. PMLR, 2019.

[15] 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. Ieee, 2009.

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

[17] Qi Dong, Shaogang Gong, and Xiatian Zhu. Imbalanced deep learning by minority class incremental rectification. *IEEE transactions on pattern analysis and machine intelligence*, 41(6):1367–1381, 2018.

[18] Michele Donini, Luca Oneto, Shai Ben-David, John Shawe-Taylor, and Massimiliano Pontil. Empirical risk minimization under fairness constraints. *arXiv preprint arXiv:1802.08626*, 2018.

[19] Simon Du, Jason Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai. Gradient descent finds global minima of deep neural networks. In *International Conference on Machine Learning*, pages 1675–1685. PMLR, 2019.

[20] John C Duchi, Peter W Glynn, and Hongseok Namkoong. Statistics of robust optimization: A generalized empirical likelihood approach. *Mathematics of Operations Research*, 2021.

[21] Vitaly Feldman. Does learning require memorization? a short tale about a long tail. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 954–959, 2020.

[22] Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimiliano Pontil. Bilevel programming for hyperparameter optimization and meta-learning. In *International Conference on Machine Learning*, pages 1568–1577. PMLR, 2018.

[23] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR, 2017.

[24] Moritz Hardt, Eric Price, and Nathan Srebro. Equality of opportunity in supervised learning. *arXiv preprint arXiv:1610.02413*, 2016.

[25] Ryuichiro Hataya, Jan Zdenek, Kazuki Yoshizoe, and Hideki Nakayama. Faster autoaugment: Learning augmentation strategies using backpropagation. In *European Conference on Computer Vision*, pages 1–16. Springer, 2020.

[26] Ryuichiro Hataya, Jan Zdenek, Kazuki Yoshizoe, and Hideki Nakayama. Meta approach to data augmentation optimization. *arXiv preprint arXiv:2006.07965*, 2020.

[27] Daniel Ho, Eric Liang, Xi Chen, Ion Stoica, and Pieter Abbeel. Population based augmentation: Efficient learning of augmentation policy schedules. In *International Conference on Machine Learning*, pages 2731–2741. PMLR, 2019.

[28] Chen Huang, Yining Li, Chen Change Loy, and Xiaoou Tang. Learning deep representation for imbalanced classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5375–5384, 2016.

[29] Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren. *Automated machine learning: methods, systems, challenges*. Springer Nature, 2019.

[30] Simon Jenni and Paolo Favaro. Deep bilevel learning. In *Proceedings of the European conference on computer vision (ECCV)*, pages 618–633, 2018.

[31] Ziwei Ji and Matus Telgarsky. Polylogarithmic width suffices for gradient descent to achieve arbitrarily small test error with shallow relu networks. *arXiv preprint arXiv:1909.12292*, 2019.

[32] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. *arXiv preprint arXiv:1910.09217*, 2019.

[33] Shuichi Katsumata and Akiko Takeda. Robust cost sensitive support vector machine. In *Artificial intelligence and statistics*, pages 434–443. PMLR, 2015.

[34] Michael Kearns. A bound on the error of cross validation using the approximation and estimation rates, with consequences for the training-test split. *Advances in Neural Information Processing Systems*, pages 183–189, 1996.

[35] Michael Kearns and Dana Ron. Algorithmic stability and sanity-check bounds for leave-one-out cross-validation. *Neural computation*, 11(6):1427–1453, 1999.

[36] Salman Khan, Munawar Hayat, Syed Waqas Zamir, Jianbing Shen, and Ling Shao. Striking the right balance with uncertainty. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 103–112, 2019.

[37] Salman H Khan, Munawar Hayat, Mohammed Bennamoun, Ferdous A Sohel, and Roberto Togneri. Cost-sensitive learning of deep feature representations from imbalanced data. *IEEE transactions on neural networks and learning systems*, 29(8):3573–3587, 2017.

[38] Mikhail Khodak, Liam Li, Maria-Florina Balcan, and Ameet Talwalkar. On weight-sharing and bilevel optimization in architecture search. 2019.

[39] Byungju Kim and Junmo Kim. Adjusting decision boundary for class imbalanced learning. *IEEE Access*, 8:81674–81685, 2020.

[40] Ganesh Ramachandra Kini, Orestis Paraskevas, Samet Oymak, and Christos Thrampoulidis. Label-imbalanced and group-sensitive classification under overparameterization. *arXiv preprint arXiv:2103.01550*, 2021.

[41] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[42] Miroslav Kubat, Stan Matwin, et al. Addressing the curse of imbalanced training sets: one-sided selection. In *Icml*, volume 97, pages 179–186. Citeseer, 1997.

[43] Hansang Lee, Minseok Park, and Junmo Kim. Plankton classification on imbalanced large scale database via convolutional neural networks with transfer learning. In *2016 IEEE international conference on image processing (ICIP)*, pages 3713–3717. IEEE, 2016.

[44] Joseph Lemley, Shabab Bazrafkan, and Peter Corcoran. Smart augmentation learning an optimal data augmentation strategy. *Ieee Access*, 5:5858–5869, 2017.

[45] Renjie Liao, Yuwen Xiong, Ethan Fetaya, Lisa Zhang, KiJung Yoon, Xaq Pitkow, Raquel Urtasun, and Richard Zemel. Reviving and improving recurrent back-propagation. In *International Conference on Machine Learning*, pages 3082–3091. PMLR, 2018.

[46] Sungbin Lim, Ildoo Kim, Taesup Kim, Chiheon Kim, and Sungwoong Kim. Fast autoaugment. *arXiv preprint arXiv:1905.00397*, 2019.

[47] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[48] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

[49] Jonathan Lorraine, Paul Vicol, and David Duvenaud. Optimizing millions of hyperparameters by implicit differentiation. In *International Conference on Artificial Intelligence and Statistics*, pages 1540–1552. PMLR, 2020.

[50] Jelena Luketina, Mathias Berglund, Klaus Greff, and Tapani Raiko. Scalable gradient-based tuning of continuous regularization hyperparameters. In *International conference on machine learning*, pages 2952–2960. PMLR, 2016.

[51] Matthew MacKay, Paul Vicol, Jon Lorraine, David Duvenaud, and Roger Grosse. Self-tuning networks: Bilevel optimization of hyperparameters using structured best-response functions. *arXiv preprint arXiv:1903.03088*, 2019.

[52] Dougal Maclaurin, David Duvenaud, and Ryan Adams. Gradient-based hyperparameter optimization through reversible learning. In *International conference on machine learning*, pages 2113–2122. PMLR, 2015.

[53] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens Van Der Maaten. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 181–196, 2018.

[54] Hamed Masnadi-Shirazi, Nuno Vasconcelos, and Arya Iranmehr. Cost-sensitive support vector machines. *arXiv preprint arXiv:1212.0975*, 2012.

[55] Aditya Menon, Harikrishna Narasimhan, Shivani Agarwal, and Sanjay Chawla. On the statistical consistency of algorithms for binary classification under class imbalance. In *International Conference on Machine Learning*, pages 603–611. PMLR, 2013.

[56] Aditya Krishna Menon, Sadeep Jayasumana, Ankit Singh Rawat, Himanshu Jain, Andreas Veit, and Sanjiv Kumar. Long-tail learning via logit adjustment. *arXiv preprint arXiv:2007.07314*, 2020.

[57] Katharina Morik, Peter Brockhausen, and Thorsten Joachims. Combining statistical learning with a knowledge-based approach: a case study in intensive care monitoring. Technical report, Technical Report, 1999.

[58] Samet Oymak, Mingchen Li, and Mahdi Soltanolkotabi. Generalization guarantees for neural architecture search with train-validation split. *arXiv preprint arXiv:2104.14132*, 2021.

[59] Samira Pouyanfar, Yudong Tao, Anup Mohan, Haiman Tian, Ahmed S Kaseb, Kent Gauen, Ryan Dailey, Sarah Aghajanzadeh, Yung-Hsiang Lu, Shu-Ching Chen, et al. Dynamic sampling in convolutional neural networks for imbalanced data classification. In *2018 IEEE conference on multimedia information processing and retrieval (MIPR)*, pages 112–117. IEEE, 2018.

[60] Saharon Rosset, Ji Zhu, and Trevor Hastie. Margin maximizing loss functions. In *NIPS*, pages 1237–1244, 2003.

[61] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019.

[62] Amirreza Shaban, Ching-An Cheng, Nathan Hatch, and Byron Boots. Truncated back-propagation for bilevel optimization. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1723–1732. PMLR, 2019.

[63] Li Shen, Zhouchen Lin, and Qingming Huang. Relay backpropagation for effective learning of deep convolutional neural networks. In *European conference on computer vision*, pages 467–482. Springer, 2016.

[64] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Joshua Susskind, Wenda Wang, and Russell Webb. Learning from simulated and unsupervised images through adversarial training. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2107–2116, 2017.

[65] Ankur Sinha, Pekka Malo, and Kalyanmoy Deb. A review on bilevel optimization: from classical to evolutionary approaches and applications. *IEEE Transactions on Evolutionary Computation*, 22(2):276–295, 2017.

[66] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR, 2019.

[67] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

[68] Robert Williamson and Aditya Menon. Fairness risk measures. In *International Conference on Machine Learning*, pages 6786–6797. PMLR, 2019.

[69] Sirui Xie, Hehui Zheng, Chunxiao Liu, and Liang Lin. Snas: stochastic neural architecture search. *arXiv preprint arXiv:1812.09926*, 2018.

[70] Zhen Xu, Andrew M Dai, Jonas Kemp, and Luke Metz. Learning an adaptive learning rate schedule. *arXiv preprint arXiv:1909.09712*, 2019.

[71] Han-Jia Ye, Hong-You Chen, De-Chuan Zhan, and Wei-Lun Chao. Identifying and compensating for feature deviation in imbalanced deep learning. *arXiv preprint arXiv:2001.01385*, 2020.

[72] Xi Yin, Xiang Yu, Kihyuk Sohn, Xiaoming Liu, and Manmohan Chandraker. Feature transfer learning for deep face recognition with under-represented data. *arXiv preprint arXiv:1803.09014*, 2018.

[73] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6023–6032, 2019.

[74] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P Gummadi. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *Proceedings of the 26th international conference on world wide web*, pages 1171–1180, 2017.

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

[76] Junjie Zhang, Lingqiao Liu, Peng Wang, and Chunhua Shen. To balance or not to balance: A simple-yet-effective approach for learning with long-tailed distributions. *arXiv preprint arXiv:1912.04486*, 2019.

[77] Zhao Zhong, Zichen Yang, Boyang Deng, Junjie Yan, Wei Wu, Jing Shao, and Cheng-Lin Liu. Blockqnn: Efficient block-wise neural network architecture generation. *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[78] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1452–1464, 2017.

[79] Boyan Zhou, Quan Cui, Xiu-Shen Wei, and Zhao-Min Chen. Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9719–9728, 2020.

[80] Pan Zhou, Caiming Xiong, Richard Socher, and Steven CH Hoi. Theory-inspired path-regularized differential network architecture search. *arXiv preprint arXiv:2006.16537*, 2020.

[81] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.

[82] Difan Zou, Yuan Cao, Dongruo Zhou, and Quanquan Gu. Stochastic gradient descent optimizes over-parameterized deep relu networks. *arXiv preprint arXiv:1811.08888*, 2018.

A Extended related works

Below we include the related work on data augmentation which was omitted from Section 2 due to space considerations.

Data Augmentation. Data augmentation techniques have been studied for decades, and many approaches such as random crop, flip, rotation, Mixup [75], Cutout [16], CutMix [73] have been applied in model training. Recently, researchers focus on automatically finding data augmentation policies to achieve better performance. Some methods [44, 64] obtain augmentation policies through an additional network or GAN. Inspired by neural architecture search, AutoAugment [12] and its folloup works [27, 46, 25, 26] formulate data augmentation as a hyper-parameter search problem. [25, 26] propose optimizing the augmentation policy using bi-level optimization by conducting differentiable relaxation on policies. Different from the above works that perform a bi-level search for an augmentation policy on a balanced dataset, our approach employs personalized data augmentation for different classes on long-tailed datasets, which leads to a better result in long-tailed learning problems.

Algorithm 2: Hyper gradient computation [49]

```
Input: Model $f_{\boldsymbol{\theta}}$ with weights $\boldsymbol{\theta}$, hyper-parameter $\boldsymbol{\alpha}$ , dataset $\mathcal{S} = \mathcal{S}_{\mathcal{T}} \cup \mathcal{S}_{\mathcal{V}}$, step sizes $\eta$, order
       of Neumann appximation $i$
$v_1 = \frac{\partial \mathcal{L}_{\text{UP}}^{\mathcal{S}_{\mathcal{V}}}}{\partial \boldsymbol{\theta}}$
$p = v_1$
for $j \leftarrow 1$ to $i$ do
    // Compute approximate Hessian inverse using Neumann series
    $v_1 = v_1(I - \eta \frac{\partial^2 \mathcal{L}_{\text{LOW}}^{\mathcal{S}_{\mathcal{T}}}}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^T})$
    $p += v_1$
end
$v_2 = -p \frac{\partial^2 \mathcal{L}_{\text{LOW}}^{\mathcal{S}_{\mathcal{T}}}}{\partial \boldsymbol{\theta} \partial \boldsymbol{\alpha}^T}$
Result: The hyper gradient $v_2$        // $v_2 = \frac{\partial \mathcal{L}_{\text{UP}}}{\partial \boldsymbol{\theta}} \left[ \frac{\partial^2 \mathcal{L}_{\text{LOW}}}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^T} \right]^{-1} \frac{\partial^2 \mathcal{L}_{\text{LOW}}}{\partial \boldsymbol{\theta} \partial \boldsymbol{\alpha}^T}$
```

B Extended experiments

In this section, we conduct additional experiments to extend the results from Sections 5 and 6. Importantly, for the experiments in Section 5 (imbalanced classes) and Section 6 (imbalanced groups), we conduct more trials (for a total of 5 trials per method) and we also provide standard error bounds in the tables. We also included additional baselines for Section 6.

Tables 5, 6, and 7 display the updated results of Section 5 for the balanced accuracy, personalized data optimization, and hyper-parameter transfer evaluation scenarios, respectively. These correspond to the original Tables 1, 2, and 3, respectively. The additional trials are generally consistent with the results and insights in the main body of the paper and demonstrates the validity of our approach.

For Section 6, we evalute two additional baselines: Distributionally Robust Optimization (DRO) [61], and a post-hoc model – described below – that tries to address group imbalance. The resuts are shown in Table 8. Overall, the results show that our Algo. 1 with $\lambda = 0.1$ consistently outperforms the other baselines. Below, we describe these two baselines in more detail.

DRO baseline: We follow the work of [61] where we optimize the DRO loss as an additional baseline. For consistency, we use a slight variation where we change the network to ResNet-18. Table 8 shows that although DRO achieves a lot better performance compared to the other baselines, our Algo. 1 with $\lambda = 0.1$ still performs noticeably better across all performance metrics.

Post-hoc baseline: We first train a ResNet-18 model with training dataset using simple CE loss. As the training dataset is imbalanced, the error of worst group is quite high at this intermediate point, more than 45%. As our posthoc model, we use vector scaling [23] which adjusts the logits. Vector scaling is essentially a generalization of Platt scaling where each logit gets its own weights in a

Method	CIFAR-10-LT	CIFAR100-LT	ImageNet-LT	iNaturalist
Cross-Entropy	30.45 ± 0.45	62.69 ± 0.16	60.82 ± 0.24	39.72 ± 0.22
LDAM loss [8]	26.37 ± 0.33	59.47 ± 0.38	58.14 ± 0.20	35.63 ± 0.21
LA loss ($\tau = 1$) [56]	23.13 ± 0.35	58.96 ± 0.20	55.57 ± 0.23	34.37 ± 0.18
Algo. 1: $\boldsymbol{\alpha} \leftarrow \tau$ of LA loss	21.82 ± 0.13	58.68 ± 0.20	55.15 ± 0.25	34.19 ± 0.19
Algo. 1: $\boldsymbol{\alpha} \leftarrow \boldsymbol{l}$	23.02 ± 0.43	58.71 ± 0.25	55.30 ± 0.29	34.35 ± 0.21
Algo. 1: $\boldsymbol{\alpha} \leftarrow \boldsymbol{\Delta}$	22.59 ± 0.26	58.40 ± 0.22	54.55 ± 0.17	34.37 ± 0.22
Algo. 1: $\boldsymbol{\alpha} \leftarrow \boldsymbol{\Delta}\&\boldsymbol{l}$	21.39 ± 0.18	56.84 ± 0.17	53.16 ± 0.17	33.41 ± 0.30
Algo. 1: $\boldsymbol{\alpha} \leftarrow \boldsymbol{\Delta}\&\boldsymbol{l}$, LA init	21.15 ± 0.22	56.70 ± 0.18	52.11 ± 0.12	33.16 ± 0.13

Table 5: Evaluations of balanced accuracy on long-tailed data with 5 trials. Algo. 1 with $\boldsymbol{\Delta}\&\boldsymbol{l}$ design space and LA initialization (bottom row) outperforms other baselines, across various datasets.

Method	CIFAR-10-LT	CIFAR100-LT	ImageNet-LT
MADAO [26]	24.42 ± 0.28	59.10 ± 0.21	59.56 ± 0.41
Algo. 1: $\boldsymbol{\alpha} \leftarrow$PDA	22.55 ± 0.38	58.52 ± 0.56	58.73 ± 0.51
Algo. 1: $\boldsymbol{\alpha} \leftarrow$PDA, $\boldsymbol{\Delta}\&\boldsymbol{l}$	20.69 ± 0.17	56.47 ± 0.23	52.18 ± 0.22

Table 6: Personalized data optimization with 5 trials.

similar fashion to the parametric cross-entropy loss. The inputs to the vector scaling post-hoc model are the output logits ($2 \times N$) from ResNet-18 where N is the sample size. We use w (2×1) and b (2×1) to adjust it $f_{\text{posthoc}}(x) = wx + b$. As there are only 4 parameters to tune, we use grid search to find the parameters that can mimimize loss (DEO or balanced error $\mathcal{E}_{\text{bal}}(f)$) on the test dataset[2]. Note that, optimizing over the test data, intuitively, makes this baseline stronger than it actually is. The posthoc model uses just 4 parameters, which can aid in balancing the result; however, as it can only adjust per class instead of per (class, group), the performance is limited compared to our approach. We note that, intuitively, our approach (or in general choosing an intelligent loss function) can be perceived as applying a posthoc adjustment during training rather than after training.

C Theoretical Insights into Pareto-Efficiency with Validation Data

In Section 4.1, we provided theoretical intuitions on why validation is necessary to build models that optimize multiple learning objectives. Within the context of this work, these objectives can be a blend of accuracy and fairness. To recap, our main intuition is that large capacity neural networks can perfectly maximize different accuracy metrics or satisfy fairness constraints such as DOE by simply fitting training data perfectly and achieving 100% training accuracy. To truly find a model that lie on the multi-objective pareto-front, the optimization procedure should (approximately) evaluate on the population landscape. Validation phase enables this as the dimensionality of the validation parameter is typically much smaller than the sample size and prevents overfitting. Below, we formalize this in a general constrained multiobjective learning setting. Suppose there are R objectives to optimize. Let $(\ell_i)_{i=1}^R$ be R loss functions and set the corresponding $\mathcal{L}_i(f) = \mathrm{E}[\ell_i(y, f(\boldsymbol{x}))]$. These can be accuracy or fairness objectives or it can even be class- or group-conditional risks (i.e. $R = K$, every class gets its own loss function).

Split $\mathcal{S} = \mathcal{T} \cup \mathcal{V}$ where $\mathcal{T}$ and $\mathcal{V}$ are training and validation respectively. During the training phase, we assume that, there is an algorithm (e.g. SGD, Adam, convex optimization, etc) A that optimizes over the training data $\mathcal{T}$ (e.g. by minimizing ERM with gradient descent). A admits the hyper-parameters $\boldsymbol{\alpha}$ (e.g., parameterization of the loss function) as input and returns a hypothesis

$$f_{\boldsymbol{\alpha}} = \mathrm{A}(\mathcal{T}, \boldsymbol{\alpha})$$

For the discussion in this section, we use $\mathcal{H}$ to denote the hyperparameter search space i.e. the values the hyperparameter $\boldsymbol{\alpha}$ can take. Let $\mathcal{L}_i^{\mathcal{V}}(f)$ be the empirical version of $\mathcal{L}_i(f)$ computed over $\mathcal{V}$. Fix penalties $\boldsymbol{\lambda} = (\lambda_i)_{i=1}^R$ which govern the combination of the loss functions (e.g. blending accuracy and fairness or weighing individual classes). The validation phase then optimizes $\boldsymbol{\theta}$ via a Multi-objective

---

[2] This is in contrast to using differentiable proxies based on cross-entropy.

Source\Target	CIFAR-10-LT	CIFAR100-LT	ImageNet-LT
CIFAR-10-LT	21.39 ± 0.18	58.10 ± 0.28	56.13 ± 0.29
CIFAR100-LT	22.40 ± 0.23	56.84 ± 0.17	55.72 ± 0.35
ImageNet-LT	23.70 ± 0.29	59.24 ± 0.23	53.16 ± 0.17

Table 7: Hyper-parameter transfer with 5 trials.

Loss function	Balanced Error	Worst (class, group) error	DEO
Cross entropy (CE)	25.37(±0.31)	46.69(±4.18)	33.75(±1.86)
CE_{bal}	21.09(±0.27)	36.63(±4.82)	20.61(±1.52)
Group-LA loss	22.91(±0.36)	40.27(±5.23)	29.34(±1.46)
CE_{DEO}	19.32(±0.31)	33.04(±5.46)	25.33(±1.35)
$0.9 \cdot \text{CE} + 0.1 \cdot \text{CE}_{\text{DEO}}$ ($\lambda = 0.1$)	20.38(±0.27)	33.36(±6.00)	26.42(±1.39)
DRO [61]	16.47(±0.23)	32.67(±3.06)	6.91(±1.30)
Posthoc: $\mathcal{E}_{\text{bal}}(f)$	21.15(±0.39)	42.83(±6.43)	32.30(±1.60)
Posthoc: $0.9 \cdot \mathcal{E}_{\text{bal}}(f) + 0.1 \cdot \mathcal{E}_{\text{DEO}}$	21.56(±0.53)	44.13(±9.39)	29.37(±2.45)
Algo. 1: with $\lambda = 0.1$	**15.50**(±0.18)	**30.33**(±2.47)	**4.25**(±0.94)

Table 8: Comparison of fairness metrics for group-imbalanced experiments, with 5 trials. Two additional baselines, DRO and post-hoc model, are evaluated.

ERM problem[3]

$$\min_{\boldsymbol{\alpha} \in \mathcal{H}} \mathcal{L}^{\mathcal{V}}_{\boldsymbol{\lambda}}(f_{\boldsymbol{\alpha}}) \quad \text{WHERE} \quad \mathcal{L}^{\mathcal{V}}_{\boldsymbol{\lambda}}(f) = \sum_{i=1}^{R} \lambda_i \mathcal{L}^{\mathcal{V}}_i(f_{\boldsymbol{\alpha}}). \tag{M-ERM}$$

Our theoretical analysis (provided below) of this setting is similar to the model-selection and cross-validation literature [34, 35, 58]. However, these works focus on a single objective. Unlike these, we will show that, small amount of validation data is enough to guarantee the pareto-efficiency of the train-validation split for all choices of $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}$. The following assumption is used by earlier work and useful for studying continuous hyperparameter spaces. The basic idea is ensuring stability of the training algorithms. This has been verified for different hyperparameter types (e.g., ridge regression parameter, continuous parameterization of the neural architecture) under proper settings [58]. We remind that, our setting is also continuous as we use differentiable optimization to determine the best loss function.

Assumption 1 (Training algorithm is stable) *Suppose $\mathcal{H} \subset \mathrm{R}^h$. There exists a partitioning of $\mathcal{H}$ into at most 2^h sets $(\mathcal{H}_i)_{i \geq 1}$ such that, over each set, the training algorithm A is locally-Lipschitz. That is, for all i and some $L > 0$, all pairs $\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2 \in \mathcal{H}_i$ and inputs $\boldsymbol{x}$ (over the support of $\mathcal{D}$) satisfies that $|f_{\boldsymbol{\alpha}_1}(\boldsymbol{x}) - f_{\boldsymbol{\alpha}_2}(\boldsymbol{x})| \leq L \|\boldsymbol{\alpha}_1 - \boldsymbol{\alpha}_2\|_{\ell_2}$.*

Here the Lipschitz constant L governs the stability level of the training algorithm. Note that, the partitioning is optional and it is included to account for discrete and discontinuous hyperparameter spaces. The following theorem shows that, if the training algorithm A satisfies stability conditions over $\mathcal{H}$ and if the validation sample size $n_{\mathcal{V}}$ is larger than R and the effective dimension of $\mathcal{H}$, then (M-ERM) does return an approximately pareto-optimal solution uniformly over all choices of $(\lambda_i, \gamma_i)_{i=1}^R$.

Theorem 1 (Multi-objective generalization) *Suppose Assumption 1 holds. Let penalties $\boldsymbol{\lambda}$ take values over the sets $\boldsymbol{\Lambda} \subset \mathrm{R}^R$. Assume the elements of the sets $\mathcal{H}, \boldsymbol{\Lambda}$ have bounded ℓ_2 norm. Suppose the loss functions have bounded derivatives (in absolute value) and, for some $\Xi > 0$, they are bounded as follows*

$$\sup_{\boldsymbol{\lambda} \in \boldsymbol{\Lambda}} \left| \sum_{i=1}^{R} \lambda_i \ell_i(y, \hat{y}) \right| \leq \Xi.$$

---

[3] We believe the results can be stated for a mixture of regularizations and constraints (e.g. enforcing the condition $\mathcal{L}^{\mathcal{V}}_i(f_{\boldsymbol{\alpha}}) \leq \tau_i$). We opted to restrict our attention to regularization in consistence with the general setting of the paper.

Given $\boldsymbol{\lambda}$*, define the corresponding* $\widehat{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{\alpha}\in\mathcal{H}} \mathcal{L}^{\mathcal{V}}_{\boldsymbol{\lambda}}(f_{\boldsymbol{\alpha}})$ *solving* (M-ERM)*. Then, with probability* $1-2e^{-t}$*, for all penalties* $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}$*, the associated* $\widehat{\boldsymbol{\alpha}}_{\boldsymbol{\lambda}}$ *achieves the population multi-objective risk*

$$\mathcal{L}_{\boldsymbol{\lambda}}(f_{\widehat{\boldsymbol{\alpha}}}) \le \arg\min_{\boldsymbol{\alpha}\in\mathcal{H}} \mathcal{L}_{\boldsymbol{\lambda}}(f_{\widehat{\boldsymbol{\alpha}}}) + \Xi\sqrt{\frac{\widetilde{\mathcal{O}}(h+R+t)}{n_{\mathcal{V}}}}. \tag{C.1}$$

Here $\widetilde{\mathcal{O}}(\cdot)$ *hides logarithmic terms. Specifically, the sample size grows only logarithmically in the stability parameter of Assumption 1 (i.e.* $\log L$ *factor).*

Interpretation: In words, this result shows that, as soon as the validation sample size is larger than $\mathcal{O}(h+R)$, train-validation split selects a model that is as good as the optimal model whose hyperparameter is tuned over the test data. That is, as $n_{\mathcal{V}}$ grows, the multi-objective risk of $f_{\widehat{\boldsymbol{\alpha}}}$ converges to risk of training with the optimal hyperparameter. In our context, it means the ability to select the optimal loss function via validation. An important remark is that, this selection happens regardless of the training phase and whether training risk overfits or not. That is, even if training returns poor models, validation phase selects the best one (out of poor options). Importantly, $h+R$ is a small number in practice. For instance, for imbalanced loss function design, h is at most $\mathcal{O}(K)$ as we use three parameters for each class. If we cluster the classes, then h is in the order of clusters. R is typically 1 (e.g. balanced accuracy in Sec 5) or 2 (e.g. DEO/accuracy tradeoffs in Sec 6). However, in the extreme case of optimizing a general combination of class-conditional risks, R can be as large as number of classes K. A remarkable aspect of this result is that, we get multi-objective pareto-efficiency of the validation-based optimization by using an extra $\mathcal{O}(R)$ samples (compared to single-loss scenario which requires $\mathcal{O}(h)$ samples [58]).

Proof The strategy is based on applying a covering argument over all variables namely $\boldsymbol{\alpha}, \boldsymbol{\lambda}$ and can be seen as a multi-objective generalization of Theorem 1 of [58]. Let $\mathcal{H}_\varepsilon, \boldsymbol{\Lambda}_\varepsilon$ be ε-covers with respect to ℓ_2-norm of the corresponding sets $\mathcal{H}, \boldsymbol{\Lambda}$. The size of these sets obey $\log|\mathcal{H}_\varepsilon| \le N_h(\varepsilon) = h\log(B/\varepsilon)$ and $\log|\boldsymbol{\Lambda}_\varepsilon| \le N_R(\varepsilon) = R\log(B/\varepsilon)$ where $B>0$ depends on the radius of $\boldsymbol{\Lambda}, \mathcal{H}$.

To proceed, pick a pair $\boldsymbol{\alpha}, \boldsymbol{\lambda}$ from the cover $\mathcal{H}_\varepsilon, \boldsymbol{\Lambda}_\varepsilon$. Define the loss function $\ell_{\boldsymbol{\lambda}} = \sum_{i=1}^R \lambda_i \ell_i(y,\hat{y})$. Since this is bounded by Ξ, we can apply Hoeffding bound for the individual cover elements. Union bounding these Hoeffding (or Ξ-sub-gaussian) concentration bounds over all cover elements, with probability $1-2e^{-t}$, we have that

$$|\mathcal{L}^{\mathcal{V}}_{\boldsymbol{\lambda}}(f_{\boldsymbol{\alpha}}) - \mathcal{L}_{\boldsymbol{\lambda}}(f_{\boldsymbol{\alpha}})| \lesssim \Xi\sqrt{\frac{(h+R)\log(B/\varepsilon)+t}{n_{\mathcal{V}}}}. \tag{C.2}$$

Perturbation analysis: To proceed, given $(\boldsymbol{\alpha}, \boldsymbol{\lambda})$ from $\mathcal{H}, \boldsymbol{\Lambda}$, choose an ε-neighboring point $(\boldsymbol{\alpha}', \boldsymbol{\lambda}')$ from the cover $\mathcal{H}_\varepsilon, \boldsymbol{\Lambda}_\varepsilon$. Let $\Gamma > 0$ be the Lipschitz constant of the loss $\ell_{\boldsymbol{\lambda}}$ (specifically over worst case $\boldsymbol{\lambda}$) and $\bar{\Xi} = \sup_{1\le r\le R}|\ell_i(y,\hat{y})|$. We note that dependence on $\Gamma, \bar{\Xi}$ will be only logarithmic. Applying triangle inequalities, we find that

$$\begin{aligned}
|\ell_{\boldsymbol{\lambda}}(y, f_{\boldsymbol{\alpha}}(\boldsymbol{x})) - \ell_{\boldsymbol{\lambda}'}(y, f_{\boldsymbol{\alpha}'}(\boldsymbol{x}))| &\le |\ell_{\boldsymbol{\lambda}}(y, f_{\boldsymbol{\alpha}}(\boldsymbol{x})) - \ell_{\boldsymbol{\lambda}}(y, f_{\boldsymbol{\alpha}'}(\boldsymbol{x}))| + |\ell_{\boldsymbol{\lambda}}(y, f_{\boldsymbol{\alpha}'}(\boldsymbol{x})) - \ell_{\boldsymbol{\lambda}'}(y, f_{\boldsymbol{\alpha}'}(\boldsymbol{x}))| \\
&\le \Gamma|f_{\boldsymbol{\alpha}'}(\boldsymbol{x}) - f_{\boldsymbol{\alpha}}(\boldsymbol{x})| + |\sum_{r=1}^R(\boldsymbol{\lambda}_i - \boldsymbol{\lambda}'_i)\ell_i(y, f_{\boldsymbol{\alpha}'}(\boldsymbol{x}))| \\
&\le \Gamma L\|\boldsymbol{\alpha}' - \boldsymbol{\alpha}\|_{\ell_2} + \bar{\Xi}\sqrt{R}\|\boldsymbol{\lambda} - \boldsymbol{\lambda}'\|_{\ell_2} \\
&\le (\Gamma L + \bar{\Xi}\sqrt{R})\varepsilon.
\end{aligned}$$

This also implies that $|\mathcal{L}_{\boldsymbol{\lambda}}(f_{\boldsymbol{\alpha}}) - \mathcal{L}_{\boldsymbol{\lambda}'}(f'_{\boldsymbol{\alpha}})|, |\mathcal{L}^{\mathcal{V}}_{\boldsymbol{\lambda}}(f_{\boldsymbol{\alpha}}) - \mathcal{L}^{\mathcal{V}}_{\boldsymbol{\lambda}'}(f'_{\boldsymbol{\alpha}})| \le (\Gamma L + \bar{\Xi}\sqrt{R})\varepsilon$. Combining this with (C.2), for all $\boldsymbol{\lambda}, \boldsymbol{\alpha}$, we obtained the uniform convergence guarantee

$$|\mathcal{L}^{\mathcal{V}}_{\boldsymbol{\lambda}}(f_{\boldsymbol{\alpha}}) - \mathcal{L}_{\boldsymbol{\lambda}}(f_{\boldsymbol{\alpha}})| \lesssim \Xi\sqrt{\frac{(h+R)\log(B/\varepsilon)+t}{n_{\mathcal{V}}}} + 2(\Gamma L + \bar{\Xi}\sqrt{R})\varepsilon.$$

Setting $\varepsilon \to \frac{\Xi}{2(\Gamma L + \bar{\Xi}\sqrt{R})\sqrt{n_{\mathcal{V}}}}$, we find that

$$|\mathcal{L}^{\mathcal{V}}_{\boldsymbol{\lambda}}(f_{\boldsymbol{\alpha}}) - \mathcal{L}_{\boldsymbol{\lambda}}(f_{\boldsymbol{\alpha}})| \lesssim \Xi \sqrt{\frac{(h+R)\log(B(\Gamma L + \bar{\Xi}\sqrt{R})\sqrt{n_{\mathcal{V}}}/\Xi) + t}{n_{\mathcal{V}}}} \tag{C.3}$$

$$\leq \Xi \sqrt{\frac{\widetilde{\mathcal{O}}(h+R+t)}{n_{\mathcal{V}}}}, \tag{C.4}$$

where we dropped the logarithmic terms. To proceed, let $\boldsymbol{\alpha}_\star$ be an optimal hyperparameter for the population risk of the validation phase i.e. $\arg\min_{\boldsymbol{\alpha}\in\mathcal{H}} \mathcal{L}_{\boldsymbol{\alpha}}(f_{\boldsymbol{\alpha}})$. We find the generalization risk of the optimal $\widehat{\boldsymbol{\alpha}}$ via

$$\mathcal{L}^{\mathcal{V}}_{\widehat{\boldsymbol{\alpha}}}(f_{\boldsymbol{\alpha}}) \leq \mathcal{L}^{\mathcal{V}}_{\boldsymbol{\alpha}_\star}(f_{\boldsymbol{\alpha}}) \leq \mathcal{L}_{\boldsymbol{\alpha}_\star}(f_{\boldsymbol{\alpha}}) + \Xi \sqrt{\frac{\widetilde{\mathcal{O}}(h+R+t)}{n_{\mathcal{V}}}},$$

concluding with the advertised result. ■

D Proof of Lemma 1

Let us recall the parametric cross-entropy loss function

$$\ell(y, f(\boldsymbol{x})) = w_y \log\left(1 + \sum_{k\neq y} e^{l_k - l_y} \cdot e^{\Delta_k f_k(\boldsymbol{x}) - \Delta_y f_y(\boldsymbol{x})}\right) = -w_y \log\left(\frac{e^{\Delta_y f_y(\boldsymbol{x}) + l_y}}{\sum_{i\in[K]} e^{\Delta_i f_i(\boldsymbol{x}) + l_i}}\right).$$

Denote the labeling likelihood $\eta_y(\boldsymbol{x}) = \mathrm{P}(y \mid \boldsymbol{x})$. When the weights w_y are not all ones, the class frequencies are effectively adjusted as $\boldsymbol{\pi}'_y \propto w_y \boldsymbol{\pi}_y$. Let $\bar{\eta}_y(\boldsymbol{x})$ be the corresponding likelihood function. The optimal score function minimizing the cross-entropy loss is given by $\Delta_y f^*_y(\boldsymbol{x}) + l_y = \log \bar{\eta}_y(\boldsymbol{x})$. This choice is determined by minimizing the expected loss (given $\boldsymbol{x}$) which sets the KL divergence between $\bar{\eta}_y(\boldsymbol{x})$ and softmax output to zero. This leads to the decision rule $f^*_y(\boldsymbol{x}) = \bar{\Delta}_y \log \frac{\bar{\eta}_y(\boldsymbol{x})}{e^{l_y}}$ where $\bar{\Delta}_y = \Delta_y^{-1}$ (to simplify the subsequent notation). Equivalently, the classification rule becomes

$$f^*_y(\boldsymbol{x}) = \log\left(\frac{\bar{\eta}_y(\boldsymbol{x})}{e^{l_y}}\right)^{\bar{\Delta}_y} \iff \text{rule}(\boldsymbol{x}) = \arg\max_{y\in[K]} \alpha_y \bar{\eta}_y^{\bar{\Delta}_y}(\boldsymbol{x}),$$

where $\alpha_y = e^{-\bar{\Delta}_y l_y}$. For standard accuracy, Bayes-optimal decision rule is $\arg\max_{y\in[K]} \bar{\eta}_y(\boldsymbol{x})$ and for balanced accuracy, it is $\arg\max_{y\in[K]} \frac{\bar{\eta}_y(\boldsymbol{x})}{\pi_y}$. In both cases, it can be written as $\arg\max_{y\in[K]} c_y \bar{\eta}_y(\boldsymbol{x})$ where c_y are adjustments.

We complete the proof by constructing a simple distribution that shows the rule($\boldsymbol{x}$) is sub-optimal. Without losing generality, we may assume $\Delta_1 \neq \Delta_2$ i.e., multiplicative adjustments of the first two classes differ. Given this multiplicative adjustment choice of $\boldsymbol{\Delta}$, we will construct a simple distribution for which minimizing parametric CE don't result in Bayes-optimal decision. Specifically, we construct input features $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ so that the first two classes (labels $y \in \{1, 2\}$) have the highest score $c_y \bar{\eta}_y(x)$ and the top two scores are close. That is, for some arbitrarily small scalars $\varepsilon, \varepsilon'$ we have $c_1 \bar{\eta}_1(\boldsymbol{x}_1) = c_2 \bar{\eta}_2(\boldsymbol{x}_1) + \varepsilon$ and $c_1 \bar{\eta}_1(\boldsymbol{x}_2) = c_2 \bar{\eta}_2(\boldsymbol{x}_2) + \varepsilon'$. Additionally, set $\bar{\eta}_i(\boldsymbol{x}_1) = \Gamma \bar{\eta}_i(\boldsymbol{x}_2)$ for $i = 1, 2$ and $\Gamma \neq 1$ an arbitrary scalar.[4] Since $\varepsilon \lesseqgtr 0$ dictates the Bayes-optimal class decision ($y = 1$ vs $y = 2$), we need the score function f^* to satisfy

$$\frac{\bar{\eta}_1(\boldsymbol{x}_1)^{\bar{\Delta}_1}}{\bar{\eta}_2(\boldsymbol{x}_1)^{\bar{\Delta}_2}} \gtreqless \frac{\alpha_2}{\alpha_1}, \quad \frac{\bar{\eta}_1(\boldsymbol{x}_2)^{\bar{\Delta}_1}}{\bar{\eta}_2(\boldsymbol{x}_2)^{\bar{\Delta}_2}} \gtreqless \frac{\alpha_2}{\alpha_1}.$$

Letting $\varepsilon, \varepsilon' \to 0$, this implies that $\frac{\bar{\eta}_1(\boldsymbol{x}_1)^{\bar{\Delta}_1}}{\bar{\eta}_2(\boldsymbol{x}_1)^{\bar{\Delta}_2}} = \frac{\bar{\eta}_1(\boldsymbol{x}_2)^{\bar{\Delta}_1}}{\bar{\eta}_2(\boldsymbol{x}_2)^{\bar{\Delta}_2}}$. However, this contradicts with the initial assumption of $\Gamma \neq 1$ via

$$\frac{\bar{\eta}_1(\boldsymbol{x}_1)^{\bar{\Delta}_1}}{\bar{\eta}_2(\boldsymbol{x}_1)^{\bar{\Delta}_2}} = \frac{\Gamma^{\bar{\Delta}_1} \bar{\eta}_1(\boldsymbol{x}_2)^{\bar{\Delta}_1}}{\Gamma^{\bar{\Delta}_2} \bar{\eta}_2(\boldsymbol{x}_2)^{\bar{\Delta}_2}} = \frac{\bar{\eta}_1(\boldsymbol{x}_2)^{\bar{\Delta}_1}}{\bar{\eta}_2(\boldsymbol{x}_2)^{\bar{\Delta}_2}} \iff \Gamma^{\bar{\Delta}_1 - \bar{\Delta}_2} = 1 \iff \Gamma = 1.$$

---

[4] Scaling the likelihoods by Γ doesn't affect $\arg\max_y c_y \bar{\eta}_y(x)$ as the other classes are assigned small probabilities.

E Proof of Lemma 2

This lemma considers the solution of the binary parametric loss defined as

$$\ell(y, f_{\boldsymbol{\theta}}(\boldsymbol{x})) = w_y \cdot \log\left(1 + e^{l_y} \cdot e^{-\Delta_y y f_{\boldsymbol{\theta}}(\boldsymbol{x})}\right).$$

Consider the ridge-constrained problem

$$\boldsymbol{\theta}_R = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{n} \ell(y_i, f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)) \quad \text{subject to} \quad \|\boldsymbol{\theta}\|_{\ell_2} \leq R. \tag{E.1}$$

The ridgeless model described in Lemma 2 obtained by minimizing the parametric loss is given by the limit $\boldsymbol{\theta}_\infty = \lim_{R\to\infty} \boldsymbol{\theta}_R / R$. Here, we focus on linear models $f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{\theta}^T \boldsymbol{x}$. The result will be established by connecting the above loss to Cost Sensitive (CS)-SVM which enforces different margins on classes. Fix $\delta > 0$. Define the CS-SVM problem as

$$\hat{\boldsymbol{w}}_\delta := \arg\min \|\boldsymbol{w}\|_2 \qquad \text{subject to} \begin{cases} \boldsymbol{w}^T \boldsymbol{x}_i \geq \delta & , y_i = 1 \\ \boldsymbol{w}^T \boldsymbol{x}_i \leq -1 & , y_i = -1 \end{cases}, \quad i \in [n]. \tag{E.2}$$

Assume the spherical data-augmentation with radii $\varepsilon_\pm$ for the two classes. Then, standard SVM on the augmented data solves

$$\min \|\boldsymbol{w}\|_2 \qquad \text{subject to} \begin{cases} \boldsymbol{w}^T \boldsymbol{x}_i - \varepsilon_+ \|\boldsymbol{w}\|_2 \geq 1 & , y_i = +1 \\ \boldsymbol{w}^T \boldsymbol{x}_i + \varepsilon_- \|\boldsymbol{w}\|_2 \leq -1 & , y_i = -1 \end{cases}, \quad i \in [n]. \tag{E.3}$$

Here, the first observation is that, since ridgeless logistic loss is equivalent to SVM[5] [60], ridgeless logistic loss with the augmented data also converges to the solution of (E.3).

For the loss function above, fix $\delta = \Delta_- / \Delta_+ > 0$. Applying Proposition 1 of [40], $\boldsymbol{\theta}_\infty$ coincides with the (ℓ_2 normalized) solution of the CS-SVM i.e.

$$\hat{\boldsymbol{w}}_\delta := \arg\min \|\boldsymbol{w}\|_2 \qquad \text{subject to} \begin{cases} \boldsymbol{w}^T \boldsymbol{x}_i \geq 1/\Delta_+ & , y_i = 1 \\ \boldsymbol{w}^T \boldsymbol{x}_i \leq -1/\Delta_- & , y_i = -1 \end{cases}, \quad i \in [n]. \tag{E.4}$$

for $\Delta_+, \Delta_- > 0$. Note that, without losing generality, we can assume $\Delta_\pm < 1$ by preserving the ratio to δ as it doesn't change the classification rule. We will prove that $\hat{\boldsymbol{w}}_\delta$ is optimal in (E.3) for the following choice of $\varepsilon_\pm$:

$$\varepsilon_\pm := \frac{1/\Delta_\pm - 1}{\|\hat{\boldsymbol{w}}_\delta\|_2}.$$

This will in turn conclude that ridgeless augmented logistic regression is equivalent to ridgeless regression with parameteric cross-entropy.

Proof of optimality of $\hat{\boldsymbol{w}}_\delta$ for (E.3). To prove the claim let $\hat{\alpha}_i,\ i \in [n]$ be the dual variables associated with (E.4) corresponding to the minimizer $\hat{\boldsymbol{w}}_\delta$. By KKT conditions it holds that $(\{\hat{\alpha}_i\}_{i\in[n]}, \hat{\boldsymbol{w}}_\delta)$ is a solution to:

$$\sum_{i\in[n]} \alpha_i y_i \boldsymbol{x}_i = \boldsymbol{w}/\|\boldsymbol{w}\|_2, \qquad \alpha_i \geq 0, \qquad \alpha_i \boldsymbol{x}_i^T \boldsymbol{w} = \begin{cases} \frac{\alpha_i}{\Delta_+} & , y_i = +1 \\ -\frac{\alpha_i}{\Delta_-} & , y_i = -1 \end{cases}, \quad i \in [n] \tag{E.5}$$

Set

$$\hat{\beta}_i = \left(\frac{1}{1 - \varepsilon_+ \sum_{i:y_i=+1} \hat{\alpha}_i - \varepsilon_- \sum_{i:y_i=-1} \hat{\alpha}_i}\right) \hat{\alpha}_i$$

With these it only takes a few algebra steps to verify that $(\{\hat{\beta}_i\}_{i\in[n]}, \hat{\boldsymbol{w}}_\delta)$ is a solution to:

$$\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|_2} - \sum_i \beta_i y_i \boldsymbol{x}_i + \varepsilon_+ \Big(\sum_{i:y_i=+1} \beta_i\Big) \frac{\boldsymbol{w}}{\|\boldsymbol{w}\|_2} + \varepsilon_- \Big(\sum_{i:y_i=-1} \beta_i\Big) \frac{\boldsymbol{w}}{\|\boldsymbol{w}\|_2} = 0 \tag{E.6}$$

$$\beta_i \geq 0, \qquad \beta_i \boldsymbol{x}_i^T \boldsymbol{w} = \begin{cases} \beta_i \big(1 + \varepsilon_+ \|\boldsymbol{w}\|_2\big) & , y_i = +1 \\ \beta_i \big(-1 - \varepsilon_- \|\boldsymbol{w}\|_2\big) & , y_i = -1. \end{cases} \tag{E.7}$$

---

[5] In the sense that ℓ_2 normalized solution of logistic regression with infinitesimal ridge is equal to the ℓ_2 normalized SVM solution.

In particular, to verify $\hat{\beta}_i \geq 0,\ i \in [n]$ we used that from the optimality of the primal-dual pair $(\{\hat{\alpha}_i\}_{i\in[n]}, \hat{\boldsymbol{w}}_\delta)$:

$$\sum_{i:y_i=+1} \frac{\hat{\alpha}_i}{\Delta_+} + \sum_{i:y_i=-1} \frac{\hat{\alpha}_i}{\Delta_-} = \|\hat{\boldsymbol{w}}_\delta\|_2,$$

and the definition of $\varepsilon_+, \varepsilon_i$. This completes the proof as it can be checked that the above corresponds exactly to the KKT conditions of (E.3).