AutoBalance: Optimized Loss Functions for Imbalanced Data

Mingchen Li **Xuechen Zhang**
University of California, Riverside
{mli176,xzhan394}@ucr.edu

Christos Thrampoulidis
University of British Columbia
cthrampo@ece.ubc.edu.ca

Jiasi Chen
University of California, Riverside
jiasi@cs.ucr.edu

Samet Oymak
University of California, Riverside
oymak@ece.ucr.edu

Abstract

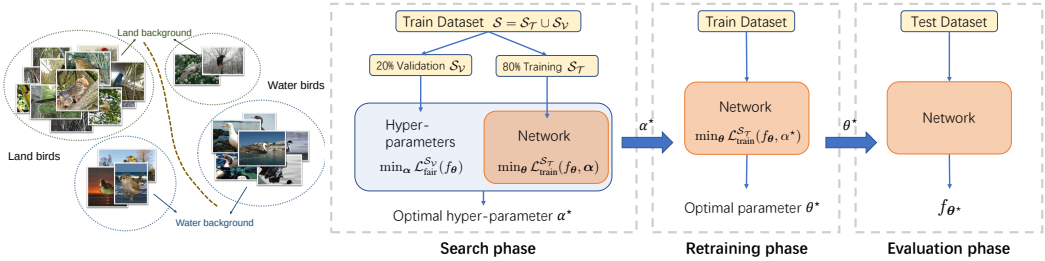
Imbalanced datasets are commonplace in modern machine learning problems. The presence of under-represented classes or groups with sensitive attributes results in concerns about generalization and fairness. Such concerns are further exacerbated by the fact that large capacity deep nets can perfectly fit 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 performance of our approach for 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 train-validation split. All code is available open-source.

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, 43, 70]. 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, 59, 25]. Indeed, 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

35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.



(a) (b)

Figure 1: (a) Example group-imbalance on the Waterbirds dataset [71, 83]. 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 $\boldsymbol{\theta}^\star$.

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 loss functions. For instance, weighted cross-entropy (e.g., each class gets a different weight, see Sec. 2) 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, 42] 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 $\ell(y, f(\boldsymbol{x})) = w_y \log\big(1 + \sum_{k \neq y} e^{l_k - l_y} \cdot e^{\Delta_k f_k(\boldsymbol{x}) - \Delta_y f_y(\boldsymbol{x})}\big)$, which allows for a personalized treatment of the individual classes via the design parameters $(w_k, l_k, \Delta_k)_{k=1}^K$ [8, 42, 59, 39, 75]. 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, 59] provide theoretically-motivated choices for (w_k, l_k), while [42] argues that (w_k, l_k) is not as effective as Δ_k in the interpolating regime of zero training error and proposes the simultaneous use of all three different parameter types. 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 [25, 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 2). We introduce novel strategies that narrow down the search space to improve convergence and avoid overfitting. To further improve the 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) [59] and label-distribution-aware margin (LDAM) [8] losses. The code is available online [47].

Algorithm 1: AutoBalance via Bilevel Optimization

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 $\ell_{\text{fair}}(\cdot) = \ell_{\text{train}}(\cdot; \boldsymbol{\alpha})$ `// Consistent initialization`

Train $\boldsymbol{\theta}$ for t_1 iterations ($\boldsymbol{\alpha}$ is fixed) `// Search Phase: Starts with warm-up`

for $i \leftarrow t_1$ **to** t_2 **do**

  Sample training batch $\mathcal{B}_{\mathcal{T}}$ from $\mathcal{S}_{\mathcal{T}}$;

  $\mathcal{B}_{\mathcal{T}} \leftarrow \mathcal{A}(\mathcal{B}_{\mathcal{T}})$ `// Apply class-personalized augmentation`

  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta_{\boldsymbol{\theta}} \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\text{train}}^{\mathcal{B}_{\mathcal{T}}}(f_{\boldsymbol{\theta}}; \boldsymbol{\alpha})$

  Sample validation batch $\mathcal{B}_{\mathcal{V}}$ from $\mathcal{S}_{\mathcal{V}}$;

  Compute hyper-gradient $\nabla_{\boldsymbol{\alpha}} \mathcal{L}_{\text{fair}}^{\mathcal{B}_{\mathcal{V}}}(f_{\boldsymbol{\theta}})$ `// via Approx. Implicit Differentiation`

  $\boldsymbol{\alpha} \leftarrow \boldsymbol{\alpha} - \eta_{\boldsymbol{\alpha}} \nabla \mathcal{L}_{\text{fair}}^{\mathcal{B}_{\mathcal{V}}}(f_{\boldsymbol{\theta}})$ `// Update loss function hyper-parameters`

end

Set $\boldsymbol{\alpha}_\star \leftarrow \boldsymbol{\alpha}$, $\mathcal{S}_{\mathcal{T}} \leftarrow \mathcal{S}$, reset weights $\boldsymbol{\theta}$ `// Retraining Phase: Use all data and` $\boldsymbol{\alpha}_\star$

Train $\boldsymbol{\theta}$ for t_2 iterations using $\boldsymbol{\alpha}_\star$

Result: The final model $\boldsymbol{\theta}_\star \leftarrow \boldsymbol{\theta}$ and hyper-parameters $\boldsymbol{\alpha}_\star$

• Extensive experiments provide several takeaways (Section 3). First, AutoBalance discovers loss functions from scratch that are consistent with theory and intuition: hyperparameters of the minority classes evolve to upweight the training loss of minority to promote them. Second, 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. Third, personalized data augmentation can further improve the performance over a single generic augmentation policy.

• Beyond class imbalance, we consider applications of loss function design to the group-sensitive setting (Section 4). Our experiments show that AutoBalance consistently outperforms various baselines, leading to a more efficient Pareto-frontier of accuracy-fairness tradeoffs.

1.1 Problem Setup for Class Imbalance

We first focus on the label-imbalance problem. The extension to the group-imbalanced setting (approach, algorithms, evaluations) is deferred to Section 4. Let $[K]$ denote the set $\{1, \ldots, K\}$. Suppose we have a dataset $\mathcal{S} = (\boldsymbol{x}_i, y_i)_{i=1}^n$ sampled i.i.d. from a distribution $\mathcal{D}$ with input space $\mathcal{X}$ and K classes. For a training example $(\boldsymbol{x}, y)$, $\boldsymbol{x} \in \mathcal{X}$ is the input feature and $y \in [K]$ is the output label. Let $f : \mathcal{X} \to \mathbb{R}^K$ be a model that outputs a distribution over classes and let $\hat{y}_f(\boldsymbol{x}) = \arg\max_{i \in [K]} f(\boldsymbol{x})$. The standard classification error is denoted by $\mathcal{E}(f) = \mathbb{P}_{\mathcal{D}}[y \neq \hat{y}_f(\boldsymbol{x})]$. For a loss function $\ell(y, \hat{y})$ (e.g. cross-entropy), we similarly denote

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

✓ **Setting: Imbalanced classes.** Define the frequency of the k'th class via $\boldsymbol{\pi}_k = \mathbb{P}_{(\boldsymbol{x},y)\sim\mathcal{D}}(y = k)$. Label/class-imbalance occurs when the class frequencies differ substantially, i.e., $\max_{i \in [K]} \boldsymbol{\pi}_i \gg \min_{i \in [K]} \boldsymbol{\pi}_i$. Let us introduce

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

Similarly, let $\mathcal{E}_{\text{k}}(f)$ be the class-conditional classification error and $\mathcal{E}_{\text{bal}}(f) := (1/K) \sum_{k=1}^{K} \mathcal{E}_{\text{k}}(f)$ be the *balanced error*. In this setting, rather than the standard test error $\mathcal{E}(f)$, our goal is to the minimize balanced error. At a high-level, we propose to do this by designing an imbalance-aware training loss that maximizes balanced validation accuracy.

2 Methods: Loss Functions, Search Space Design, and Bilevel Optimization

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 [50, 39, 17] to optimize balanced objectives. Our design space will utilize recent works which introduce Label-distribution-aware margin (LDAM) [8], Logit-adjustment (LA) [59], Class-dependent temperatures (CDT) [75], Vector scaling (VS) [42] losses. Specifically, we

build on the following parametric loss function controlled by three vectors $\boldsymbol{w}, \boldsymbol{l}, \boldsymbol{\Delta} \in \mathbb{R}^K$:

$$\ell(y, f(\boldsymbol{x})) = w_y \log\big(1 + \sum_{k \neq y} e^{l_k - l_y} \cdot e^{\Delta_k f_k(\boldsymbol{x}) - \Delta_y f_y(\boldsymbol{x})}\big). \tag{2.1}$$

Here, w_y enables conventional weighted CE and l_y, and Δ_y are additive and multiplicative adjustments to the logits. This choice is same as the VS-loss introduced in [42], which borrows the $\boldsymbol{\Delta}$ term from [75] and $\boldsymbol{l}$ term from [8, 59]. [59] makes the observation that we can use $\boldsymbol{l}$ rather than $\boldsymbol{w}$ while ensuring Fisher consistency in balanced error. We make the following complementary observation.

Lemma 1 *Parametric loss function* (2.1) *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, [42] 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 (2.1) 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:

$$\ell_{\text{train}}(y, \boldsymbol{x}, f; \boldsymbol{\alpha}) = -\,\mathbb{E}_{\mathcal{A}}\left[w_y \log\left(\frac{e^{\sigma(\Delta_y) f_y(\mathcal{A}_y(\boldsymbol{x})) + l_y}}{\sum_{i \in [K]} e^{\sigma(\Delta_i) f_i(\mathcal{A}_y(\boldsymbol{x})) + l_i}}\right)\right]. \tag{2.2}$$

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

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. The PDA idea relates to SMOTE [9], where the minority class is over-sampled by creating synthetic examples. To formalize the benefits of PDA, 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 [42]. The following intuitive observation links the PDA to parametric loss (2.1).

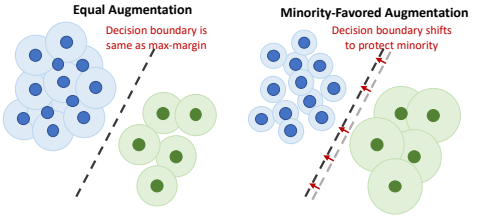


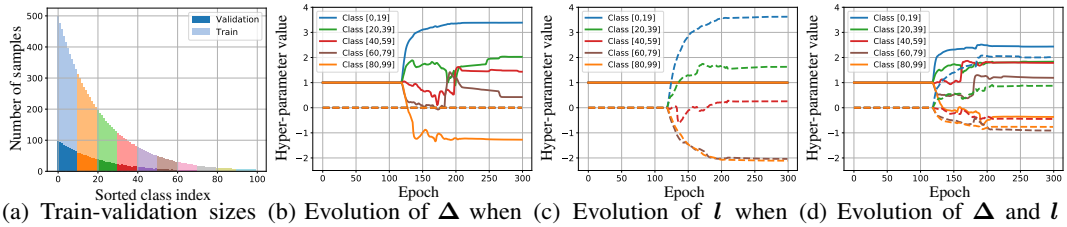
Figure 2: Data augmentation can shift the decision boundary to benefit the minority class by providing a larger margin. Lemma 2 establishes an equivalence between spherical data augmentation and parametric cross-entropy loss.

Lemma 2 *Consider a binary classification task with labels* 0 *and* 1 *and a linearly separable training dataset. For any parametric loss* (2.1) *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*** (2.1)**.**

This lemma is similar in flavor to [35], which considers a larger uncertainty set around the minority class. But, as discussed in the appendix, Lemma 2 is relevant in the overparameterized regime whereas the approach of [35] is ineffective for separable data [57]. Algorithmically, the augmentations that we consider are much more flexible than the ℓ_p-balls of [35] and our experiments showcase the value of our approach in state-of-the-art multiclass settings. Besides, note that the (theoretical) benefits of PDA can go well-beyond Lemma 2 by leveraging the invariances [10, 14] (via rotation, translation).

2.1 Proposed Bilevel Optimization Method

We formulate the loss function design as a bilevel optimization over hyperparameters $\boldsymbol{\alpha}$ and a hypothesis set $\mathcal{F}$. We split the dataset $\mathcal{S}$ into training $\mathcal{S}_{\mathcal{T}}$ and validation $\mathcal{S}_{\mathcal{V}}$ sets with $n_{\mathcal{T}}$ and $n_{\mathcal{V}}$



(a) Train-validation sizes for CIFAR100-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 CIFAR100-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.

examples respectively. Let $\mathcal{E}_{\text{fair}}$ be the desired test-error objective. When $\mathcal{E}_{\text{fair}}$ is not differentiable, we use a weighted cross-entropy (CE) loss function $\ell_{\text{fair}}(y, \hat{y})$ chosen to be consistent with $\mathcal{E}_{\text{fair}}$. For instance, $\mathcal{E}_{\text{fair}}$ could be a superposition of standard and balanced classification errors, i.e. $\mathcal{E}_{\text{fair}} = (1-\lambda)\mathcal{E} + \lambda\mathcal{E}_{\text{bal}}$. Then, we simply choose $\ell_{\text{fair}} = (1-\lambda)\text{CE} + \text{CE}_{\text{bal}}$. The hyperparameter $\boldsymbol{\alpha}$ aims to minimize the loss ℓ_{fair} over the validation set $\mathcal{S}_{\mathcal{V}}$ and the hypothesis $f \in \mathcal{F}$ aims to minimize the training loss (2.2) as follows:

$$\min_{\boldsymbol{\alpha}} \mathcal{L}_{\text{fair}}^{\mathcal{S}_{\mathcal{V}}}(f_{\boldsymbol{\alpha}}) \quad \text{WHERE} \quad f_{\boldsymbol{\alpha}} = \arg\min_{f \in \mathcal{F}} \mathcal{L}_{\text{train}}^{\mathcal{S}_{\mathcal{T}}}(f; \boldsymbol{\alpha}) := \frac{1}{n_{\mathcal{T}}} \sum_{i=1}^{n_{\mathcal{T}}} \ell_{\text{train}}(y_i, \boldsymbol{x}_i, f; \boldsymbol{\alpha}). \tag{2.3}$$

Here, $\mathcal{L}_{\text{fair}}/\mathcal{L}_{\text{fair}}^{\mathcal{S}_{\mathcal{V}}}$ are the test/validation risks associated with ℓ_{fair}, e.g. $\mathcal{L}_{\text{fair}} = \mathbb{E}_{\mathcal{D}}[\ell_{\text{fair}}]$ as in Section 1.1. Algorithm 1 summarizes our approach and highlights the key components. The training loss $\ell_{\text{train}}(\cdot; \boldsymbol{\alpha})$ is also initialized to be consistent with $\mathcal{E}_{\text{fair}}$ (e.g., same as ℓ_{fair}). In line with the literature on bilevel optimization, we will refer to the two minimizations of the validation and training losses in (2.3) as upper and lower level optimizations, respectively.

Implicit Differentiation and Warm-up training. For a loss function parameter $\boldsymbol{\alpha}$, the hyper-gradient can be written via the chain-rule $\frac{\partial \mathcal{L}_{\text{fair}}(\boldsymbol{\theta}^\star)}{\partial \boldsymbol{\alpha}} = \frac{\partial \mathcal{L}_{\text{fair}}}{\partial \boldsymbol{\alpha}} + \frac{\partial \mathcal{L}_{\text{fair}}}{\partial \boldsymbol{\theta}^\star} \frac{\partial \boldsymbol{\theta}^\star}{\partial \boldsymbol{\alpha}}$ [52]. Here, $\boldsymbol{\theta}^\star$ is the solution of the lower-level problem. We note that $\partial \mathcal{L}_{\text{fair}}/\partial \boldsymbol{\alpha} = 0$ since $\boldsymbol{\alpha}$ does not appear within the upper-level loss. Also observe that $\partial \mathcal{L}_{\text{fair}}(\boldsymbol{\theta}^\star)/\partial \boldsymbol{\theta}^\star$ can be directly computed by taking the gradient. To compute $\partial \boldsymbol{\theta}^\star/\partial \boldsymbol{\alpha}$, we follow the recent work [52] and employ the Implicit Function Theorem (IFT). If there exists a fixed point $(\boldsymbol{\theta}^\star, \boldsymbol{\alpha}^\star)$ that satisfies $\partial \mathcal{L}_{\text{train}}(\boldsymbol{\theta}^\star, \boldsymbol{\alpha}^\star)/\partial \boldsymbol{\theta} = 0$ 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 $\frac{\partial \boldsymbol{\theta}}{\partial \boldsymbol{\alpha}} = (\frac{\partial^2 \mathcal{L}_{\text{train}}}{\partial \boldsymbol{\theta}^2})^{-1} \frac{\partial^2 \mathcal{L}_{\text{train}}}{\partial \boldsymbol{\theta} \partial \boldsymbol{\alpha}}$. However, directly computing inverse Hessian $(\frac{\partial^2 \mathcal{L}_{\text{train}}}{\partial \boldsymbol{\theta}^2})^{-1}$ 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 [48, 52]. Finally, the warm-up phase of our method (Line 2 of Algo. 1) is essential to guarantee that the IFT assumption $\frac{\partial \mathcal{L}_{\text{train}}(\boldsymbol{\theta}^\star, \boldsymbol{\alpha}^\star)}{\partial \boldsymbol{\theta}} = 0$ is approximately satisfied.

Why Bilevel Optimization? We choose differentiable optimization over alternative hyperparameter tuning methods because our hyperparameter space is continuous and potentially large (e.g. in the order of $K = 8,142$ for iNaturalist). In our experiments, the runtime of our method was typically 4$\sim$5 times that of standard training (with known hyperparameters). Intuitively, the runtime is at least twice due to our use of separate search and retraining phases. In the appendix, we also compare against alternative approaches (specifically SMAC of [30]) and found that our approach is faster and more accurate on CIFAR10-LT.

2.2 Reducing the Hyperparameter Search Space and the Benefits of Validation Set

Suppose we wish to optimize the hyperparameter $\boldsymbol{\alpha} = (w_y, l_y, \Delta_y)_{y=1}^K$ of the parametric loss (2.1). An important challenge is the dimensionality of $\boldsymbol{\alpha}$, which is proportional to the number of classes K, as we need a triplet (w_y, l_y, Δ_y) for each class. For instance, ImageNet has $K = 1,000$

whereas iNaturalist has $K = 8,142$ classes resulting in high-dimensional hyperparameters. In our experiments, we found that directly optimizing over such large spaces leads to convergence issues likely because of the difficulty of hypergradient estimation. Additionally, with large number of hyperparameters there is increased concern for validation overfitting. This is especially so for the tail classes (e.g. the smallest class in CIFAR100-LT has only 1 validation example with an 80-20% split). On the other hand, it is well-known in AutoML literature (e.g. neural architecture search [51], AutoAugment [12]) that designing a good search space is critical for attaining faster convergence and good validation accuracy. To this end, we propose *subspace-based search spaces* for hyperparameters (w_y, l_y, Δ_y). To explain the idea, consider the logit-adjustment parameters $\boldsymbol{l} = [l_1 \ \dots \ l_K]$ and $\boldsymbol{\Delta} = [\Delta_1 \ \dots \ \Delta_K]$. We propose representing these K dimensional vectors via $K' < K$ dimensional embeddings $\boldsymbol{l}', \boldsymbol{\Delta}'$ as follows

$$\boldsymbol{\Delta} = \boldsymbol{D}_{\boldsymbol{\pi}} \boldsymbol{\Delta}' \quad \text{and} \quad \boldsymbol{l} = \boldsymbol{D}_{\boldsymbol{\pi}} \boldsymbol{l}' \qquad \text{where} \qquad \boldsymbol{l}', \boldsymbol{\Delta}' \in \mathbb{R}^{K'}.$$

Here, $\boldsymbol{D}_{\boldsymbol{\pi}} \in \mathbb{R}^{K \times K'}$ is a *frequency-aware dictionary* matrix that we design, and the range space of $\boldsymbol{D}_{\boldsymbol{\pi}}$ becomes the hyperparameter search space. In our algorithm, we cluster the classes in terms of their frequency and assign the same hyperparameter to classes with similar frequencies. To be concrete, if each cluster has size C, then $K' = \lceil K/C \rceil$. For CIFAR10-LT, CIFAR100-LT, ImageNet-LT and iNaturalist, we use $C = \{1, 10, 20, 40\}$ respectively. In this scheme, each column $\boldsymbol{d}$ of the matrix $\boldsymbol{D}_{\boldsymbol{\pi}}$ is the indicator function of one of the K' clusters, i.e. $\boldsymbol{d}_i = 1$ if ith class is within the cluster and 0 otherwise. Clusters are pictorially illustrated in Fig. 3a. Finally, we remark that the specific hyperparameter choices of CDT [75], LA [59], and VS [42] losses in the corresponding papers, can be viewed as specific instances of the above search space design. For instance, LA-loss chooses a scalar τ and sets $l_i = \tau \log(\boldsymbol{\pi}_i)$. This corresponds to a dictionary containing a single column $\boldsymbol{d} \in \mathbb{R}^K$ with entries $\boldsymbol{d}_i = \log(\boldsymbol{\pi}_i)$.

Why is train-validation split critical? In Figure 4 we plot the balanced errors of training/validation/test datasets at each epoch of the search phase. The training data is fixed whereas we evaluate different validation set sizes. The first finding is that training loss always overfits (dotted) until zero error whereas validation loss mildly overfits (dashed vs solid). Secondly, larger validation does help improve test accuracy (compare solid lines). The training behavior is in line with the fact that large capacity networks can perfectly fit and achieve 100% training accuracy [19, 62, 33]. This also means that different accuracy metrics or fairness constraints can be perfectly satisfied. To truly find a model that lies on the Pareto-front of the (accuracy, fairness) tradeoff, the optimization procedure should (approximately) evaluate on the population loss. Thus, as in Figure 4, the validation phase provides this crucial test-proxy in the overparameterized setting where training error is vacuous. Following the model selection literature [36, 37], 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. Our search space design in Section 2.2 also helps to this end by increasing the oversampling ratio $n_{\mathcal{V}}/\dim(\boldsymbol{\alpha})$ via class clustering. In the appendix, we formalize these intuitions for multi-objective problems (e.g., accuracy + fairness). Under mild assumptions, 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 hyperparameter $\boldsymbol{\alpha}$ minimizing the validation risk $\mathcal{L}_{\text{fair}}^{\mathcal{S}_{\mathcal{V}}}(f) = (1-\lambda)\mathcal{L}_1^{\mathcal{S}_{\mathcal{V}}} + \lambda \mathcal{L}_2^{\mathcal{S}_{\mathcal{V}}}$ in (2.3) also approximately minimizes the test risk $\mathcal{L}_{\text{fair}}(f)$.

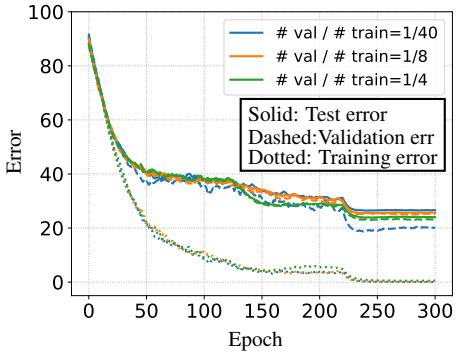


Figure 4: Train/Validation/Test errors during CIFAR10-LT search phase with different validation sizes and fixed training size.

3 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 $\mathcal{L}_{\text{fair}}$ is the balanced cross-entropy

Method	CIFAR10-LT	CIFAR100-LT	ImageNet-LT	iNaturalist
Cross-Entropy	30.45	62.69	55.47	39.72
LDAM loss [8]	26.37	59.47	54.21	35.63
LA loss ($\tau = 1$) [59]	23.13	58.96	52.46	34.06
CDT loss [75]	**20.73**	57.26	53.47	34.46
AutoBalance: τ of LA loss	21.82	58.68	52.39	34.19
AutoBalance: $\boldsymbol{l}$	23.02	58.71	52.60	34.35
AutoBalance: $\boldsymbol{\Delta}$	22.59	58.40	53.02	34.37
AutoBalance: $\boldsymbol{\Delta}\&\boldsymbol{l}$	21.39	56.84	51.74	33.41
AutoBalance: $\boldsymbol{\Delta}\&\boldsymbol{l}$, LA init	21.15	**56.70**	**50.91**	**33.25**

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 most of the baselines, across various datasets.

CE_{bal}. We consider various designs for ℓ_{train} 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 [59, 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 **CIFAR10-LT** and **CIFAR100-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 [59]. 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 [52]), we start the validation optimization after 120 epochs of the training optimization, using SGD with initial learning rate 0.05, momentum 0.9, and weight decay $1e-4$, we follow the same learning rate decay at epoch 220 and 260. For CIFAR10-LT, 20 hyper-parameters are trained, corresponding to l_y and Δ_y of each 10 classes. For CIFAR100-LT, ImageNet-LT, and iNaturalist, we reduce the search space with cluster sizes of 10, 20, and 40 as visualized in Figure 3(a) and as described in Section 2.2. For ImageNet-LT and iNaturalist, following previous work [59], we use ResNet-50 and SGD for the lower and upper optimizations, For the learning rate scheduling, we use cosine scheduling starting with learning rate 0.05, and batch size 128. In searching phase, we conduct 150 epoch training with 40 epoch warm-up before the loss function design starts. For the retraining phase, we train for 90 epochs, which is the same as [59] but only due to the lack of training resources we change the batch size to 128 and adjust initial learning rate accordingly as suggested by [23].

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, [27]); 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 (2.1). 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 $\tau = 1$ [59]. 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 CIFAR10-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 [27] 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 CIFAR10-LT). More importantly, the last two lines of the table demonstrates that PDA can be synergistically combined with the parametric CE (2.1) which leads to further improvements, however, we observe that most of the improvement can be attributed to (2.1).

Method	CIFAR10-LT	CIFAR100-LT	ImageNet-LT
MADAO [27]	24.39	59.10	55.31
AutoBalance: PDA	22.53	58.55	54.47
AutoBalance: $\boldsymbol{\Delta}\&\boldsymbol{l}$	21.39	56.84	51.74
AutoBalance: PDA, $\boldsymbol{\Delta}\&\boldsymbol{l}$	20.76	56.49	51.50

Table 2: The evaluations on personalized data optimization.

4 Approaches and Evaluations for Imbalanced Groups

While Section 3 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 group-imbalanced setting.

✓ **Setting: Imbalanced groups.** For the setting with G groups, dataset is given by $\mathcal{S} = (\boldsymbol{x}_i, y_i, g_i)_{i=1}^n$ where $g_i \in [G]$ is the group-membership. In the fairness literature, groups represent sensitive or protected attributes. For $(\boldsymbol{x}, y, g) \sim \mathcal{D}$, define the group and (class, group) frequencies as

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

The group-imbalance 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 $\max_{(k,j)} \boldsymbol{\pi}_{k,j} \gg \min_{(k,j)} \boldsymbol{\pi}_{k,j}$. 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) [25, 18]. Our evaluations also focus on binary classification (with labels denoted via $\pm$) and two groups ($K = G = 2$). With this setup, the DEO risk is defined as $\mathcal{L}_{\text{deo}}(f) = |\mathcal{L}_{+,1}(f) - \mathcal{L}_{+,2}(f)|$. Here $\mathcal{L}_{k,j}(f)$ is the (class, group)-conditional risk evaluated on the conditional distribution of "Class k & Group j". When both classes are equally relevant (rather than $y = +1$ implying a semantically positive outcome), we use the symmetric DEO:

$$\mathcal{L}_{\text{deo}}(f) = |\mathcal{L}_{+,1}(f) - \mathcal{L}_{+,2}(f)| + |\mathcal{L}_{-,1}(f) - \mathcal{L}_{-,2}(f)|. \tag{4.1}$$

We will study the pareto-frontiers of the DEO (4.1), group-balanced error, and standard error. Here, group-balanced risk is defined as $\mathcal{L}_{\text{bal}}^{\mathcal{G}}(f) = \frac{1}{KG}\sum_{k=1}^{K}\sum_{j=1}^{G} \mathcal{L}_{k,j}(f)$. Note that this definition treats each (class, group) pair as its own (sub)group. Throughout, we explicitly set the validation loss to cross-entropy for clarity, thus we use CE, $\text{CE}_{\text{bal}}^{\mathcal{G}}$, CE_{deo} to refer to $\mathcal{L}$, $\mathcal{L}_{\text{bal}}^{\mathcal{G}}$, $\mathcal{L}_{\text{deo}}$.

Validation (upper-level) loss function. In Algo. 1, we set $\mathcal{L}_{\text{fair}} = (1 - \lambda_{\text{val}}) \cdot \text{CE} + \lambda_{\text{val}} \cdot \text{CE}_{\text{deo}}$ for varying $0 \leq \lambda_{\text{val}} \leq 1$. The parameter λ_{val} enables a trade-off between accuracy and fairness. Within λ_{val}, we use the subscript *"val"* to highlight the fact that we regularize the validation objective rather than the training objective.

Group-sensitive training loss design. As first proposed in [42], the parametric cross-entropy (CE) can be extended to (class, group) imbalance by extending hyper-parameter $\boldsymbol{\alpha}$ to $[K] \times [G]$ variables $\boldsymbol{w}, \boldsymbol{l}, \boldsymbol{\Delta} \in \mathbb{R}^{[K]\times[G]}$ generalizing (2.1), (2.2). This leads us to the following parametric loss function for group-sensitive classification [1]

$$\ell_{\text{train}}(y, g, f(\boldsymbol{x}); \boldsymbol{\alpha}) = -w_{yg} \log\left(\frac{e^{\sigma(\Delta_{yg})f_y(\boldsymbol{x})+l_{yg}}}{\sum_{k\in[K]} e^{\sigma(\Delta_{kg})f_k(\boldsymbol{x})+l_{kg}}}\right). \tag{4.2}$$

Here, w_{yg} applies weighted CE, while Δ_{yg} and l_{yg} are logit adjustments for different (class, groups). This loss function is used throughout the imbalanced groups experiments.

[1] This is a generalization to multiple classes of the proposal in [42] for binary group-sensitive classificaiton.

Loss function	Balanced Error	Worst (class, group) error	DEO
Cross entropy (CE)	23.38	43.25	33.75
$\text{CE}^{\mathcal{G}}_{\text{bal}}$	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
DRO [65]	16.47	32.67	6.91
AutoBalance: $\mathcal{L}_{\text{fair}}$ with $\lambda_{\text{val}} = 0.1$	**15.13**	**30.33**	**4.25**

Table 3: Comparison of fairness metrics for group-imbalanced experiments. The first six rows are different training loss choices, where $\text{CE}^{\mathcal{G}}_{\text{bal}}$, Group-LA, CE_{deo}, and DRO 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 $\lambda_{\text{val}} = 0.1$ in Algo. 1.

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 3 and Figure 5, 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 $w_g = 1/\bar{\boldsymbol{\pi}}_g$ to balance group frequencies and then applies logit-adjustment with $\tau = 1$ over the classes conditioned on the group-membership.

Datasets. We experiment with the modified Waterbird dataset [65]. 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 [65], 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 [65], 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 3.

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 5, we show the influence of the parameter λ where $\mathcal{L}_{\text{reg}}$ is chosen to be CE_{deo}, $\text{CE}^{\mathcal{G}}_{\text{bal}}$, 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.

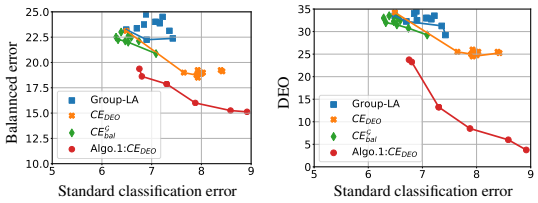


Figure 5: Waterbirds fairness-accuracy tradeoffs for parametrized loss designs $(1-\lambda)\cdot\text{CE}+\lambda\cdot\mathcal{L}_{\text{reg}}$, for different $\mathcal{L}_{\text{reg}}$ choices. Group-balanced error $\mathcal{E}^{\mathcal{G}}(f)$ (left) and DEO $\mathcal{E}_{\text{deo}}(f)$ (right) are 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 3, we solely focus on optimizing the fairness objectives (rather than standard error). Thus, we compare against CE_{deo}, $\text{CE}^{\mathcal{G}}_{\text{bal}}$, Group-LA as the baseline approaches as well as blending CE with $\lambda = 0.1$. We also compare to the DRO approach of [65]. Finally, we display the outcome of Algo. 1 with $\lambda_{\text{val}} = 0.1$. While DRO is competitive, similar to Figure 5, our approach outperforms all baselines for all metrics. The improvement is particularly significant when it comes to DEO. Finally, we remark that [42] further proposed combining DRO with the group-adjusted VS-loss for improved performance. We leave the evaluation of AutoBalance for such combinations to future.

5 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 [44, 67, 9, 45, 1, 81, 63, 76, 5, 56]. Several loss re-weighting schemes [60, 58, 29, 13, 6, 38] have been proposed to adjust weights of different classes or samples during training. Another line of work [80, 41, 59] focuses on post-hoc correction. More recently, several works [50, 39, 17, 38, 8, 13, 59, 75] develop more refined class-balanced loss functions (e.g. (2.1)) that better adapt to the training data. In addition, several works [34, 84] 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. Group-sensitive learning aims to ensure fairness in the presence of under-represented groups (e.g., gender, race). [7, 25, 78, 72] 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 [25, 18] (also recall DEO (4.1)). [65] discusses group-sensitive learning in an over-parameterized regime and proposes that strong regularization ensures fairness. Closer to our work, [65, 42] also study Waterbirds dataset. Compared to the regularization-based approach of [65], we explore a parametric loss design (inspired by [42]) 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 4.

Bilevel Optimization. Classical approaches [69] for hyper-parameter optimization are typically based on derivative-free schemes, including random search [73] and reinforcement learning [86, 3, 74, 82]. Recently, a growing line of works focus on differentiable algorithms that are often faster and can scale up to millions of parameters [52, 55, 66, 32, 54]. These techniques [51, 40, 85, 53] have shown significant success in neural architecture search, learning rate scheduling, regularization, etc. They 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, 61]. 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).

6 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) [31]. 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 [59, 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 [59]). Second, to improve generalization and avoid overfitting to validation, we reduce the hyperparameter search space by grouping the classes with similar frequencies. Finally, evaluations show that the designed loss functions are interpretable (Fig. 3(b,c,d)) and are consistent with theoretical intuitions.

Acknowledments

This work is supported in part by the National Science Foundation under grants CCF-2046816, CNS-1932254, CCF-2009030, HDR-1934641, CNS-1942700 and by the ARO under W911NF-21-1-0312.

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] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

[24] 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.

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

[26] 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.

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

[28] 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.

[29] 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.

[30] Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown. Sequential model-based optimization for general algorithm configuration. In *International conference on learning and intelligent optimization*, pages 507–523. Springer, 2011.

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

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

[33] 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.

[34] 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.

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

[36] 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.

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

[38] 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.

[39] 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.

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

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

[42] Ganesh Ramachandra Kini, Orestis Paraskevas, Samet Oymak, and Christos Thrampoulidis. Label-imbalanced and group-sensitive classification under overparameterization. *accepted to the Thirty-fifth Conference on Neural Information Processing Systems (NeurIPS)*, 2021.

[43] 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.

[44] 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.

[45] 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.

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

[47] Mingchen Li, Xuechen Zhang, Christos Thrampoulidis, Jiasi Chen, and Samet Oymak. Autobalance source code. https://github.com/ucr-optml/AutoBalance, 2021.

[48] 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.

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

[50] 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.

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

[52] 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.

[53] 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.

[54] 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.

[55] 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.

[56] 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.

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

[58] 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.

[59] 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.

[60] 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.

[61] Samet Oymak, Mingchen Li, and Mahdi Soltanolkotabi. Generalization guarantees for neural architecture search with train-validation split. *International Conference on Machine Learning*, 2021.

[62] Samet Oymak and Mahdi Soltanolkotabi. Towards moderate overparameterization: global convergence guarantees for training shallow neural networks. *IEEE Journal on Selected Areas in Information Theory*, 2020.

[63] 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.

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

[65] 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.

[66] 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.

[67] 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.

[68] 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.

[69] 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.

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

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

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

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

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

[75] 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.

[76] 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.

[77] 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.

[78] 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.

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

[80] 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.

[81] Yuan Zhao, Jiasi Chen, and Samet Oymak. On the role of dataset quality and heterogeneity in model confidence. *ICML Workshop on Uncertainty & Robustness in Deep Learning*, 2020.

[82] 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.

[83] 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.

[84] 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.

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

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