Appendix Outline

. This appendix is organized as follows:

- In Appendix A, we provide more detailed results on additional datasets for the different components discussed in Section 4. Specifically, in subsections Appendices A.1 to A.7, we present results on batch size, data augmentation, model architectures, pre-training, SSL, Sharpness-Aware Minimization, and label smoothing. These subsections delve into the specific effects and outcomes of each component.
- In Appendix A.8, we examine the relationship between the training and test distributions in imbalanced training. We explore the optimal balance of training data and discuss the potentially destructive impact of collecting additional majority samples.
- In Appendix A.9, we present additional and extended experimental results that compare the methods proposed in our paper with the baseline methods.
- In Appendix A.10, we provide additional experimental results that illustrate how the training process evolves for imbalanced data.
- In Appendix A.11, we include decision boundary visualizations for imbalanced training. Specifically, we demonstrate that Sharpness-Aware Minimization (SAM-A) helps decision regions take up similar volumes, whereas standard training routines tend to shrink-wrap the decision boundaries around minority samples.
- In Appendix B, we provide detailed information on the hyperparameters, datasets, and architectures used in our experiments.
- In Appendix C, we discuss the limitations of our study. This section addresses potential constraints, challenges, and areas for improvement in our research.
- Lastly, in Appendix D, we discuss the broader impact of our work. This section explores the implications, significance, and potential applications of our findings beyond the scope of the immediate study.

A Additional Experiments

A.1 Batch Size

To investigate the impact of batch size in the context of class imbalance, we train networks across various training ratios using different batch sizes. In order to compare the accuracy for each training ratio, we calculate the percentage improvement over the baseline (set as the best batch size of 128). Specifically, if we denote Acc_b^{ρ} as the accuracy on the imbalanced dataset with training ratio ρ and batch size b, we define the adjusted accuracy $new_{A}cc_b^{\rho}$ as

$$\bar{A}cc_b^{\rho} = \frac{Acc_b^{\rho} - Acc_{128}^{\rho}}{Acc_b^{\rho}}. \tag{1}$$

Positive values represent higher accuracy compared to the baseline, while negative values denote lower accuracy. This normalization allows us to examine the relative effect of batch size. As shown in the main text and Figure 6, data with a high degree of class imbalance tends to benefit from smaller batch sizes, despite the fact that small batches often do not contain any minority samples.

A.2 Data Augmentation

In order to evaluate and compare the effectiveness of various popular augmentation techniques—including horizontal flips, random crops, AugMix [28], TrivialAugmentWide [49], and AutoAugment [11]—we investigate their impact on the accuracy of minority and majority classes across a range of training ratios.

We measure the relative improvement in performance by comparing the accuracy achieved with data augmentation to that achieved without it. We thus plot the percentage improvement as a function of the training ratio in Figure 7.

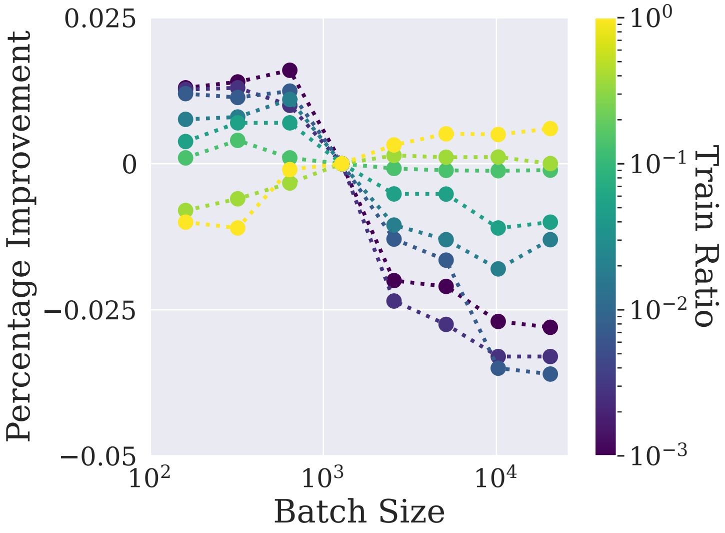


Figure 6: Batch size matter more for imbalanced data where small batch sizes are best, whereas the curve corresponding to balanced data is flat. Percentage improvement in test accuracy over the default batch size of 128 at different training ratios. Experiments conducted on CIFAR-10.

Our findings reveal that while the newer TrivialAugment method exhibits superior performance on balanced training data, the older AutoAugment method yields better results on highly imbalanced data.

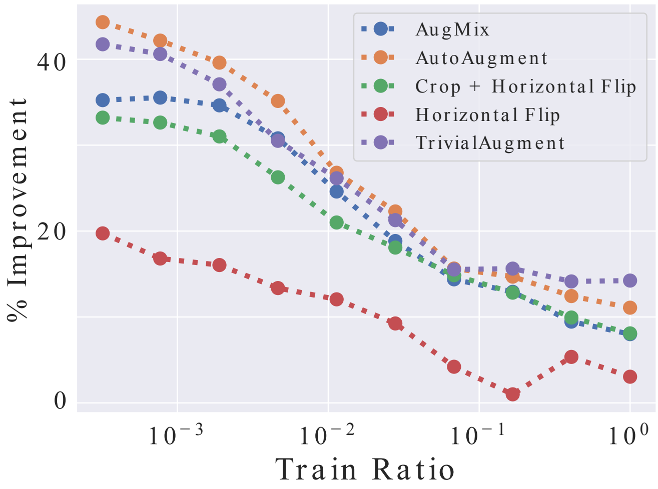


Figure 7: Optimal augmentations depend on the imbalance ratio. We plot the percent improvement in test accuracy for different augmentations compared to training without augmentations across train ratios for different augmentations. We see that TrivialAugment, which is known to outperform AutoAugment on class-balanced data, actually performs worse when data is severely imbalanced. Experiments conducted on CIFAR-100.

A.3 Model architecture

In Figure 9, we illustrate the impact of model size on the performance of the CIFAR-10 dataset with a training ratio of 0.001. The trend observed is similar to the results discussed in the main text, where increasing the model size leads to overfitting in the case of imbalanced training.

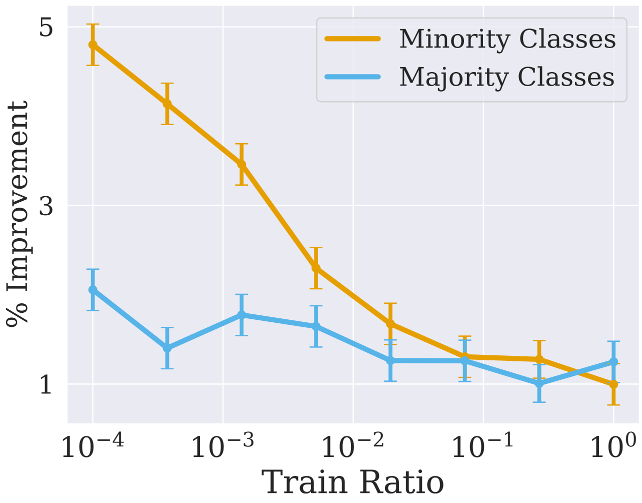


Figure 8: Strong augmentations are particularly effective at improving minority class accuracy under severe class imbalance. The percent improvement in test accuracy of TrivialAugment compared to training without any augmentation as a function of the training ratio. Experiments conducted on CIFAR-10. Error bars represent one standard error over 5 trials.

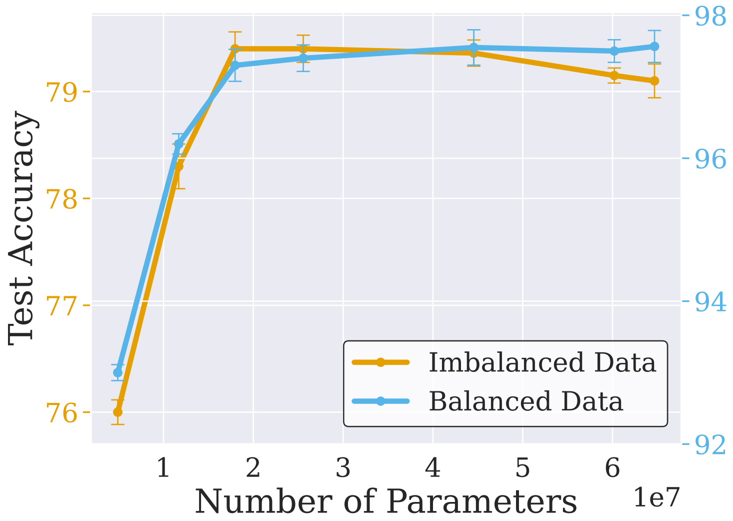


Figure 9: Bigger architectures overfit on class-imbalanced data. Experiments conducted on CIFAR-10. Error bars represent one standard error over 5 trials.

A.4 Pre-training

To assess the effectiveness of pre-training, we fine-tune several pre-trained models on downstream datasets with varying training ratios. In addition to the main body, Figure 11 illustrates the percentage improvement in test accuracy compared to random initialization for supervised pre-training on ImageNet-1k and ImageNet-21k, as well as SimCLR on ImageNet-1k (which is a Self-Supervised Learning (SSL) method), measured by downstream performance on CIFAR-10. This comparison is made across different training ratios (Figure 4). Let $\mathrm{Acc}^{\rho}_{\text{Rand}}$ denote the accuracy of the model trained from random initialization at a training ratio ρ. The relative improvement is then defined by:

$$\bar{A}cc_b^{\rho} = \frac{Acc_b^{\rho} - Acc_{\text{Rand}}^{\rho}}{Acc_{\text{Rand}}^{\rho}} \tag{2}$$

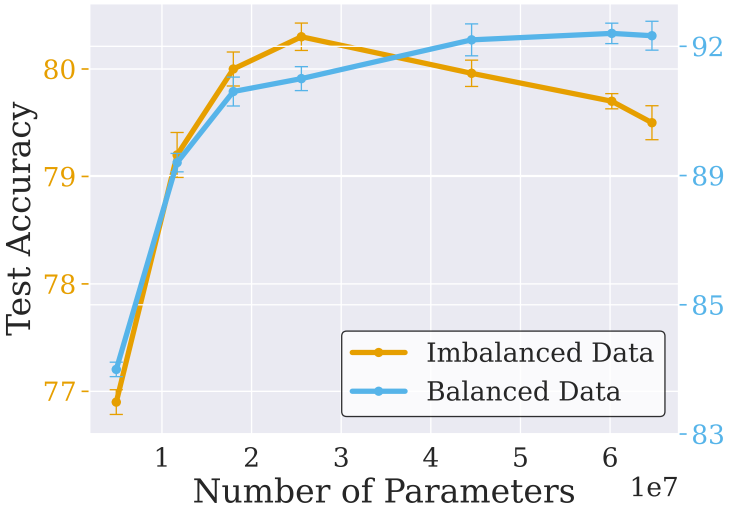


Figure 10: Bigger architectures overfit on class-imbalanced data. Experiments were conducted on CINIC-10 with an imbalanced train ratio of 0.001. Error bars represent one standard error over 5 trials.

Positive values indicate an improvement in performance compared to random initialization. It is clear that all pre-training methods improve performance when compared to random initialization. Interestingly, these improvements are significantly more pronounced under imbalanced conditions.

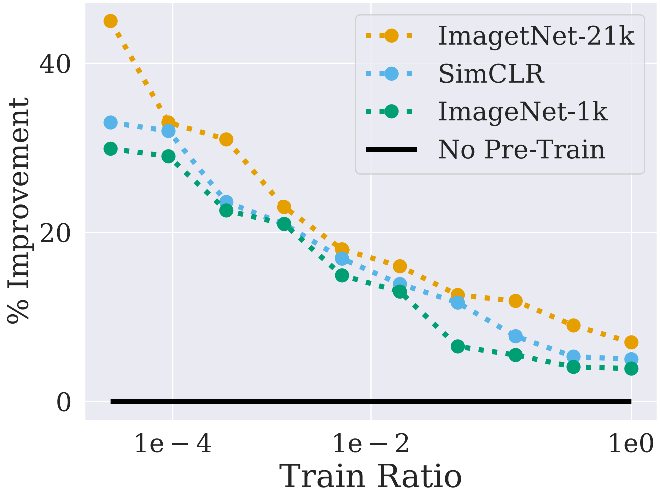


Figure 11: Pretraining yields bigger improvements on more imbalanced data. The improvement in the test accuracy compared to training from random initialization. Experiments conducted on CIFAR-10.

A.5 SSL

Self-supervised learning (SSL) has gained substantial traction as a method of representation learning across multiple domains, including computer vision, natural language processing, and tabular data [10, 36, 55]. Networks pretrained using SSL often demonstrate more transferable representations than those pretrained with supervision [21]. Pre-training traditionally consists of a two-stage process: initial learning on an upstream task followed by fine-tuning on a downstream task. However, the limitation in many use-cases is the lack of large-scale pretraining datasets. In order to solve this problem, our approach diverges from this two-stage process by merging supervised learning with an auxiliary self-supervised loss function during from-scratch training, effectively eliminating the need for any pertaining.

For this, we employ the Variance-Invariance-Covariance Regularization (VICReg) objective [4]:

Given two batches of embeddings, $\boldsymbol{Z} = [f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_B)]$ and $\boldsymbol{Z}' = [f(\boldsymbol{x}'_1), \ldots, f(\boldsymbol{x}'_B)]$, each of size $(B \times K)$, where $\boldsymbol{x}_i$ and $\boldsymbol{x}'_i$ are two distinct random augmentations of a sample I_i, we derive the covariance matrix $\boldsymbol{C} \in \mathbb{R}^{K \times K}$ from $[\boldsymbol{Z}, \boldsymbol{Z}']$.

Consequently, the VICReg loss can be articulated as:

$$\mathcal{L}_{SS\mathcal{L}} = \frac{1}{K} \sum_{k=1}^{K} \left(\alpha \max\left(0, \gamma - \sqrt{\boldsymbol{C}_{k,k} + \epsilon}\right) + \beta \sum_{k' \neq k} (\boldsymbol{C}_{k,k'})^2 \right) + \gamma \|\boldsymbol{Z} - \boldsymbol{Z}'\|_F^2 / N.$$

In our experiments, the total loss is given by

$$L_{Joint-SSL} = L_{SSL} + \lambda L_{\text{Supervised}}.$$

Note that the SSL loss function is independent of the class-imbalanced labels.

A.6 SAM

Sharpness-Aware Minimization [18] is an optimization technique that seeks to find "flat" minima of the loss function, often leading to improved generalization. This method consists of taking an initial ascent step followed by a descent step, aiming to find parameters that minimize the increase in loss resulting from the ascent step. Huang et al. [33] demonstrate that flat minima correspond to wide-margin decision boundaries.

Given a model parameterized by weights θ and a loss function $L(\theta)$ that we aim to minimize, SAM performs two steps in each iteration:

1. **First step (gradient ascent):** Perform a scaled gradient ascent step from the current model weights θ:

$$\theta' = \theta + \rho |\nabla L(\theta)|_2 \frac{\nabla L(\theta)}{|\nabla L(\theta)|_2} \tag{3}$$

2. **Second step (weight update):** Update the weights from θ in the negative direction of the gradient computed at the post-ascent parameter vector:

$$\theta = \theta - \eta \nabla L\left(\theta'\right) \tag{4}$$

In the above steps, η represents the learning rate, ρ is a hyperparameter determining the size of the neighborhood around the current weights, and $|\cdot|_2$ denotes the Euclidean norm.

SAM was initially developed for balanced datasets, where the decision boundaries for each class have comparable areas. However, this assumption does not hold true for imbalanced datasets. To address this, we adapted SAM for use with class-imbalanced datasets by increasing the flatness specifically for minority class loss terms. We propose a new method - SAM-Asymmetric (SAM-A). Our method adjusts the ascent step size (ρ) in SAM's inner loop for minority classes by employing a step size inversely proportional to the classes' proportions.

Let p_i be the proportion of class i in the training set. We define the class-conditional ascent step size as:

$$\rho_i = \frac{\rho}{1 - p_i}, \tag{5}$$

where ρ is a scaling factor.

By doing this, we widen the margins around under-represented classes, potentially improving generalization in imbalanced datasets.

A.7 Label Smoothing

Label smoothing is a regularization technique often used in training deep learning models. It mitigates the model's excessive confidence in class labels, which can improve generalization and reduce overfitting. However, traditional label smoothing assumes a balanced class distribution, which is not always the case in real-world datasets.

To adapt label smoothing for imbalanced training, we propose a class-conditional label smoothing technique. Instead of using a uniform smoothing parameter ϵ, we use a different ϵ_i for each class i, which is proportional to the inverse of the class's proportion within the dataset.

Let p_i be the proportion of class i in the training set. We define the class-conditional smoothing parameter as:

$$\epsilon_i = \frac{\epsilon}{1 - p_i}, \tag{6}$$

where ϵ is a scaling factor.

We then apply label smoothing as follows. Let p be the model's output probability distribution over K classes, and let q_i be the target distribution for class i. The smoothed target distribution is:

$$q_{i,j} = (1 - \epsilon_i) I_{y=j} + \frac{\epsilon_i}{K}, \tag{7}$$

where $j \in 1, 2, ..., K$, y is the true class, and $I_{.}$ is the indicator function.

During training, we minimize the cross-entropy loss between the model's predictions p and the class-conditional smoothed labels q_i:

$$L = -\sum_{i=1}^{K} q_{i,y} \log p_y \tag{8}$$

By using class-conditional label smoothing, we apply more smoothing to the minority classes and less to the majority classes, which can help the model generalize better when the class distribution is imbalanced.

A.8 Data Curation

Common intuition dictates that training on data that is more balanced than the testing distribution can improve representation learning by preventing overfitting to minority samples [22, 8, 24]. In this section, we put that intuition to the test by examining the optimal balance of training data. Moreover, while minority class samples may be scarce, a practitioner may be able to collect additional majority class training samples at will, so we also examine the potentially destructive impact of collecting additional majority samples.

A.8.1 The Relationship Between Train-Time and Test-Time Imbalance

The literature on training routines for class imbalance in machine learning is filled with methods designed for scenarios in which training data is highly imbalanced but testing data is balanced. However, data encountered during deployment is typically also imbalanced. Therefore, we disentangle training and testing balances and investigate how sensitive models are to discrepancies between the two. This study may be particularly important if one considers collecting training data for a downstream application. Should we gather training data with the same balance we anticipate during testing? How worried should we be if the data we encounter during deployment is more or less balanced than the training data we gathered?

We begin by illustrating three scenarios in Figure 12: (1) identical training and testing ratios, (2) balanced training, and (3) the training ratio with the lowest test error (optimal training ratio). We see that training on data with the same imbalance as the testing data is superior to training on balanced data, and the two strategies only approach equal performance when the testing data becomes balanced. We share additional results over different datasets and models in Figure 20, Figure 21, and Figure 22.

We then plot for each test ratio the corresponding train ratio that results in the lowest test error in Figure 13. If the two ratios are perfectly aligned, then points will lie on the diagonal. Indeed, the points are close to the diagonal, indicating that it is best to train with a very similar imbalance ratio to the test dataset, especially for highly imbalanced testing scenarios.

In these previous experiments, we fixed the size of the training set, but what happens as we gather more and more training data? In Figure 16, we train and evaluate a network on different imbalance ratios across training set sizes, and we plot the misalignment between the train and test ratios, referring to the average distance between the optimal train ratio and the specified test ratio. As the amount of training data increases, we see that the optimal training ratio becomes more and more close to the ratio of the test data.

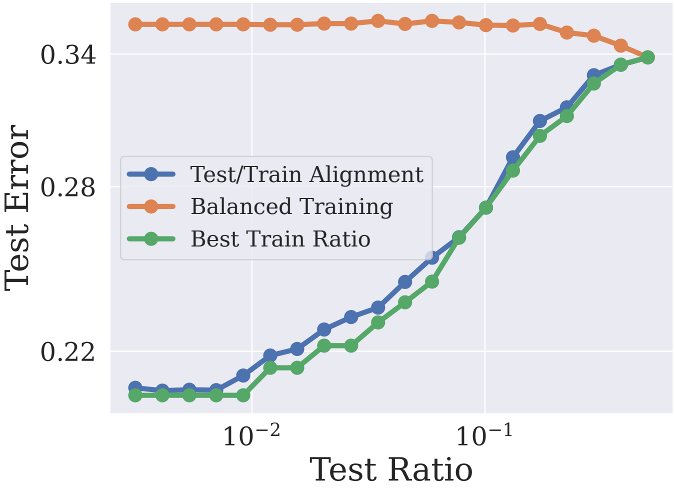


Figure 12: Imbalanced training data is optimal for imbalanced testing scenarios. Test accuracy as a function of the test ratio for different training setups. Experiments conducted on CIFAR-100.

A.8.2 When More Data Degrades Performance

In practice, a practitioner may not have precise control over the data they collect. Will collecting additional samples always help performance? Instead of fixing the total number of samples and

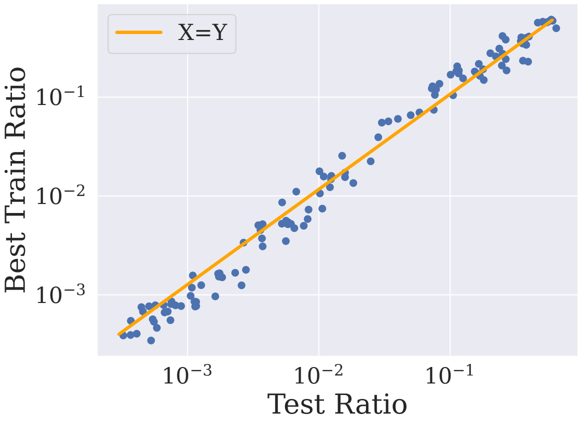


Figure 13: The optimal train ratio is closely aligned with the test ratio. Experiments conducted on CIFAR-100.

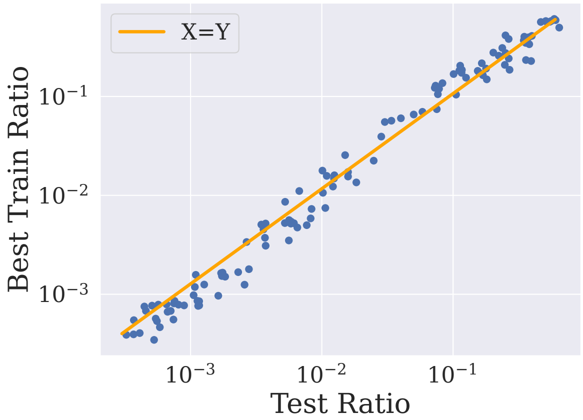


Figure 14: The optimal train ratio is closely aligned with the test ratio. Experiments conducted on CINIC-10.

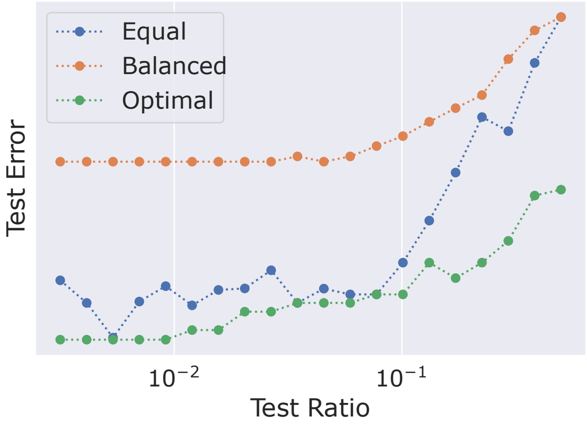


Figure 15: Test accuracy on the minority classes as a function of the test ratio for different training setups. 'Equal' denotes the same balance between training and testing, and 'Optimal' is the optimal trainset balance amongst the ratios we try. Experiments conducted on CIFAR-10.

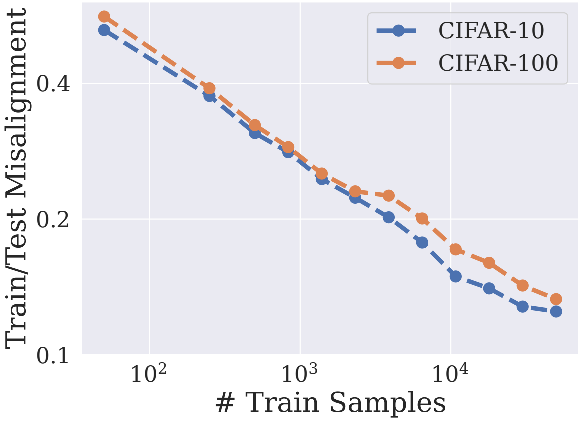


Figure 16: Alignment between train and test proportions improves as the number of training samples increases. *Train/test misalignment* is calculated by taking the mean over test ratios of the difference between the best train ratio (train ratio that gives maximum test accuracy) and the test ratio. If misalignment is 0, then the best train ratio is always the same as the test ratio.

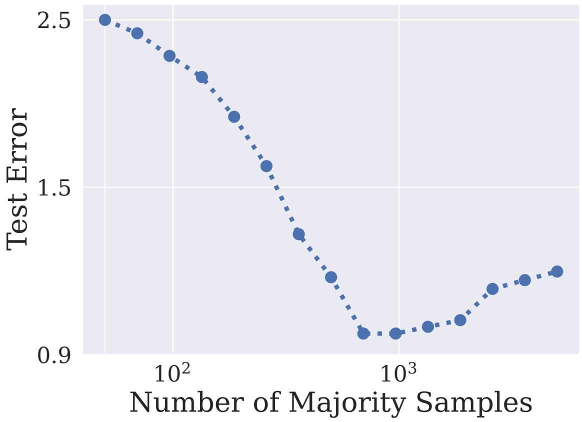


Figure 17: The potentially destructive effects of adding majority class samples. We fix the number of minority samples to be 200 and vary the number of majority samples. Experiments conducted on CIFAR-100.

Table 5: Our training routines exceed previous state-of-the-art or improve existing methods when combined. Split class accuracy for classes with Few, Med and Many examples of WideResNet-28×10 on long-tailed CIFAR-100 and CINIC-10. Error bars correspond to one standard error over 5 trials.

	CINIC-10			CIFAR-100		
Method	Few	Med	Many	Few	Med	Many
ERM	40.5 ± 0.4	64.1 ± 0.3	90.1 ± 0.5	20.1 ± 0.3	42.3 ± 0.3	70.5 ± 0.6
Reweighting	36.6 ± 0.5	63.1 ± 0.3	87.8 ± 0.3	17.1 ± 0.4	39.3 ± 0.3	67.1 ± 0.4
Resampling	37.4 ± 0.5	63.6 ± 0.6	87.9 ± 0.4	18.4 ± 0.2	38.1 ± 0.2	68.9 ± 0.3
Focal Loss	39.1 ± 0.2	63.9 ± 0.2	88.2 ± 0.5	19.8 ± 0.4	39.0 ± 0.5	69.3 ± 0.6
LDAM-DRW	40.1 ± 0.4	64.3 ± 0.4	89.8 ± 0.3	20.8 ± 0.5	42.1 ± 0.3	70.6 ± 0.4
M2m	42.8 ± 0.7	64.1 ± 0.6	90.3 ± 0.4	20.1 ± 0.6	41.8 ± 0.4	69.4 ± 0.5
SAM-A	43.2 ± 0.3	62.3 ± 0.6	89.7 ± 0.3	22.5 ± 0.4	40.3 ± 0.3	70.1 ± 0.4
Joint-SSL + SAM-A	43.9 ± 0.4	63.3 ± 0.5	90.4 ± 0.5	22.9 ± 0.3	41.3 ± 0.6	69.9 ± 0.6
Joint-SSL + SAM-A + M2m	44.1 ± 0.3	64.2 ± 0.4	90.9 ± 0.3	23.9 ± 0.4	42.3 ± 0.2	70.4 ± 0.3

varying their imbalance ratio, we now fix the number of samples from the minority class and vary the number of total majority class samples.

In Figure 17, we see that increasing the number of samples from the majority class initially boosts performance on a balanced test set. Nevertheless, in both cases, the performance reaches an optimum before the growing training data imbalance eventually degrades test accuracy. Thus, adding training data can help, but if we add enough majority samples, we must be careful not to cause too sharp a mismatch between training and testing distributions. Notably, the optimal training set ratio is nearly balanced, matching the test set, even when we are allowed to gather extra samples from one class without having to forego samples from another.

A.9 Benchmarking Results

In Table 5, we present additional experimental results that compare the methods proposed in our paper with the baseline methods. In accordance with Kang et al. [34], we also report the accuracy across three distinct subsets: (1) Many-shot classes, which contain more than 100 training samples. (2) Medium-shot classes, comprising 20 to 100 samples, and (3) Few-shot classes, including classes with fewer than 20 samples.

A.10 Regularization and Overfitting

In order to determine whether the performance differences among various methods stem from their optimization abilities or their generalization to unseen test samples, we evaluate the training error without any regularization or specialized optimization method. Specifically, we train a ResNet-50 network on CIFAR-10 and CIFAR-100 datasets using SGD with an initial learning rate of 0.5 and cosine annealing, across different levels of training data imbalance. As seen in Figure 18, although fitting all training examples takes longer as we increase the imbalance ratio of our datasets, the empirical risk minimization successfully fits all training data eventually, including minority samples.

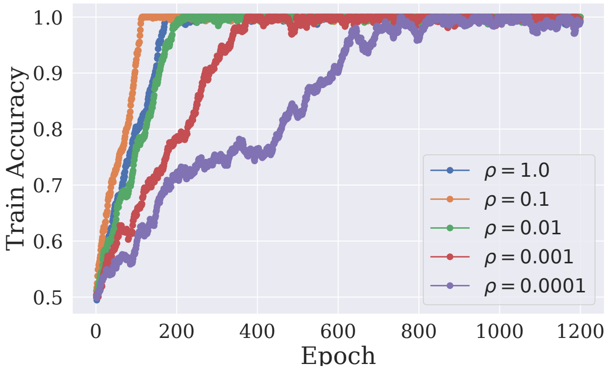


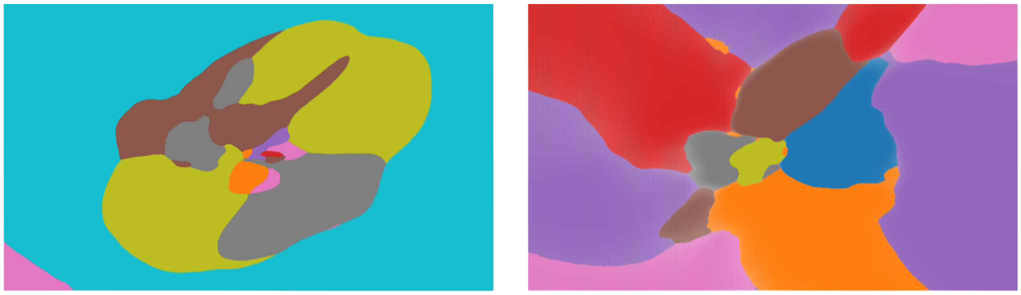
Figure 18: Imbalanced data is harder to fit. Training accuracy every epoch for imbalanced training with various imbalance ratios. Experiments conducted on CIFAR-10.

A.11 Decision Boundary Visualizations

To explore the differences between classifiers trained on imbalanced data, we visualize their decision boundaries. A variety of methods have been established for visualizing the decision boundaries of deep learning models, offering valuable insights into their intricate internal operations. Apart from the methods discussed in the main text, we utilize the approach introduced by Somepalli et al. [56] to visualize the decision boundaries of a ResNet-50 network trained on the CIFAR-10 dataset. In Figure 19, we display the decision boundaries resulting from standard training (right), which yields narrow margins around minority classes (green, grey, and orange), and SAM-A (left), which notably broadens these margins and all the classes occupy similar area in input space.

B Experimental Details

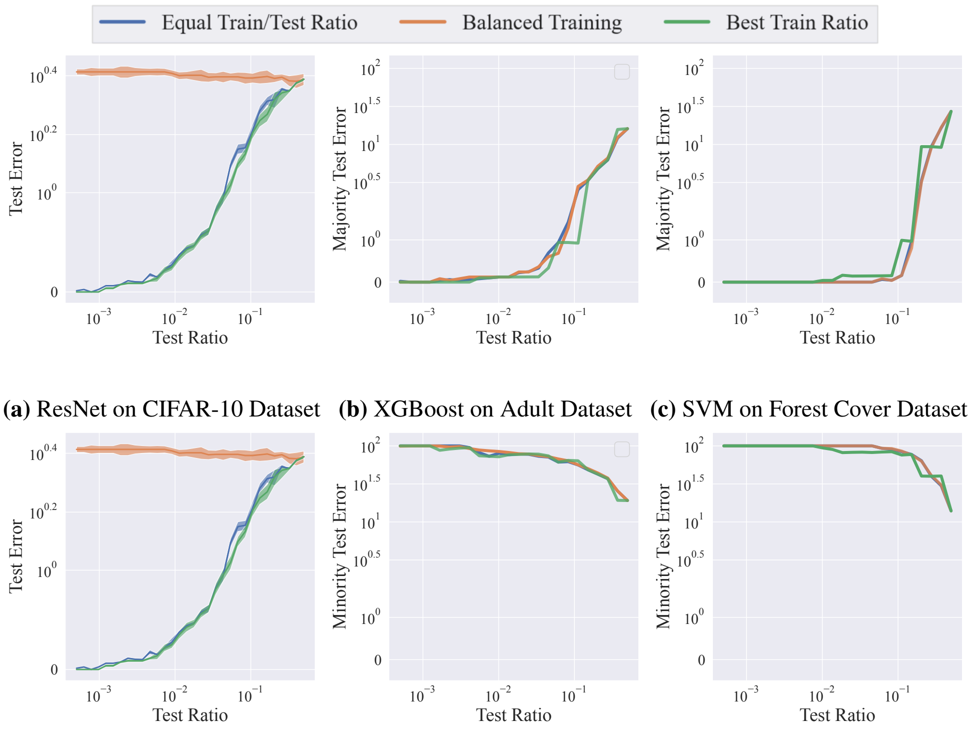
In this section, we provide additional implementation details that were not included in the main text.



(a) After naive training

(b) After SAM training

Figure 19: SAM-A makes decision regions take up similar volumes, whereas standard training routines shrink wrap the decision boundaries around minority samples. Experiments conducted on a CIFAR-10 with ResNet-18.



(a) ResNet on CIFAR-10 Dataset **(b)** XGBoost on Adult Dataset **(c)** SVM on Forest Cover Dataset

(d) ResNet on CIFAR-10 Dataset **(e)** XGBoost on Adult Dataset **(f)** SVM on Forest Cover Dataset

Figure 20: Test error split by majority and minority classes for balanced test sets. We see similar trends across all models and datasets.

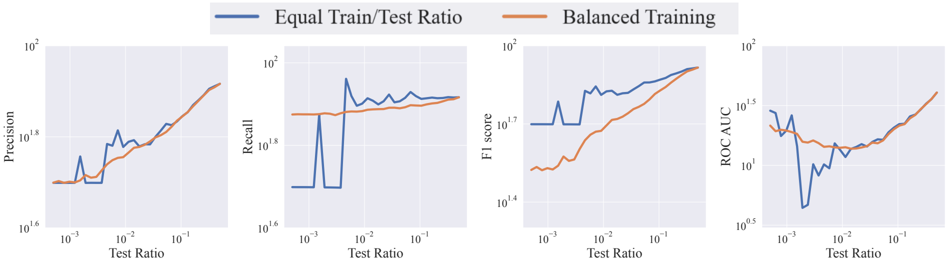


Figure 21: Additional metrics for XGBoost on the Adult dataset.

For the CIFAR-10, CIFAR-100, and CINIC-10 datasets, we follow the imbalanced setup proposed by Liu et al. [45], using an exponential distribution to create imbalances between classes. Across all methods, we use TrivialAugment [49] combined with CutMix as our augmentation policy, supplemented by label smoothing and an exponential moving weight average. Our model of choice is the WideResNet-28×10 [62].

We employ the SGD optimizer with momentum 0.9 and weight decay coefficient 210^{-4}. Our models are trained for 300 epochs with cosine annealing and a linear warm-up of the learning rate. The learning rate is initialized at 0.1.

For the APTOS 2019 Blindness Detection, SIIM-ISIC Melanoma Classification, and EuroSAT datasets, we largely follow the approach detailed in Fang et al. [17], utilizing the ResNeXt-50-32×4d model, which was identified as the best model for these datasets in the comparison by Fang et al. [17].

Our implementation was done in PyTorch, utilizing the PyTorch Lightning library for training. All of our models were trained on V100 GPUs.

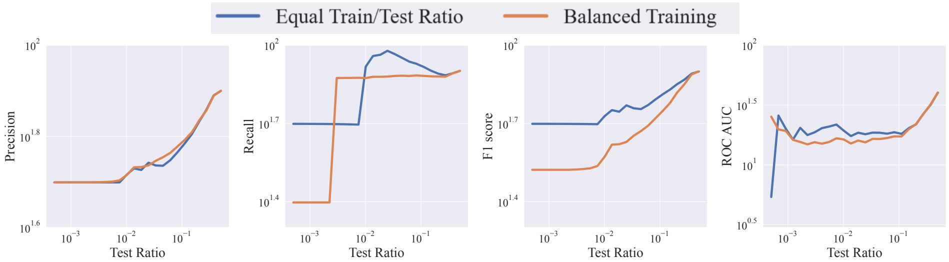


Figure 22: Additional metrics for SVM on the Forest Cover dataset.

C Limitations

In our paper, we found that existing methods for class imbalance are unreliable on real-world datasets. While our tuned routine was effective on the real-world datasets we considered, these general trends raise the concern that solutions which are effective on some class-imbalanced datasets may fail on others. A second limitation of our work is that some tools we utilize are only applicable in certain domains. For example, data augmentations and self-supervised learning for tabular data are not widely accepted.

D Broader Impacts

Across a wide variety of high-impact domains, ranging from credit card fraud detection to disease diagnosis, data is severely class-imbalanced. Therefore, performance increases for class-imbalanced data is highly valuable. With this potential for value also comes the potential that proposed methods make false promises which won't benefit real-world practitioners and may in fact cause harm when deployed in sensitive applications. For this reason, we release our numerical results across diverse datasets, and we also include implementation details for the sake of transparency and reproducibility. As with all new state-of-the-art methods, our improvements may also improve models used for malicious intentions.