Constrained Optimization to Train Neural Networks on Critical and Under-Represented Classes

Anonymous Author(s)
Affiliation
Address
email

Abstract

Deep neural networks (DNNs) are notorious for making more mistakes for the classes that have substantially fewer samples than the others during training. Such class imbalance is ubiquitous in clinical applications and very crucial to handle because the classes with fewer samples most often correspond to critical cases (e.g., cancer) where misclassifications can have severe consequences. Not to miss such cases, binary classifiers need to be operated at high True Positive Rates (TPR) by setting a higher threshold but this comes at the cost of very high False Positive Rates (FPR) for problems with class imbalance. Existing methods for learning under class imbalance most often do not take this into account. We argue that prediction accuracy should be improved by emphasizing reducing FPRs at high TPRs for problems where misclassification of the positive, i.e., critical, class samples are associated with higher cost. To this end, we pose the training of a DNN for binary classification as a constrained optimization problem and introduce a novel constraint that can be used with existing loss functions to enforce maximal area under the ROC curve (AUC) through prioritizing FPR reduction at high TPR. We solve the resulting constrained optimization problem using an Augmented Lagrangian method (ALM). Going beyond binary, we also propose two possible extensions of the proposed constraint for multi-class classification problems. We present experimental results for image-based binary and multi-class classification applications using an in-house medical imaging dataset, CIFAR10, and CIFAR100. Our results demonstrate that the proposed method improves the baselines in majority of the cases by attaining higher accuracy on critical classes while reducing the misclassification rate for the non-critical class samples.[1]

1 Introduction

Deep Neural Networks (DNNs) perform extremely well in many classification tasks when sufficiently large and representative datasets are available for training. However, in many real world applications, it is not uncommon to encounter highly-skewed class distributions, i.e., majority of the data belong to only a few classes while some classes are represented with scarce instances. Training DNNs on such imbalanced datasets leads to models that are biased toward majority classes with poor prediction accuracy for the minority class samples. While this is problematic for all such applications, it poses an even greater issue for "critical" applications where misclassifying samples belonging to the minority class can have severe consequences. One domain where such applications are common and machine learning is having an important impact is medical imaging.

In medical imaging, applications with data imbalance are ubiquitous [14] and cost of making some types of mistakes are more severe than others. For instance, in a diagnosis application, discarding a

[1] Code is available in the supplementary material.

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

cancer case as healthy (False Negative) is more costly than classifying a healthy subject as having cancer (False Positive). While the latter creates burden for the subject as well as health-care system through additional tests that may be invasive and expensive, the former, i.e., failure to identify a cancerous case, would delay the diagnosis and jeopardise treatment success. In such applications, binary classifiers are operated at high True Positive Rates (TPR) even when this means having higher False Positive Rates (FPR). To make matters more complicated, there are usually significantly fewer samples to represent critical classes where mistakes are more severe. For instance, in [6] authors found out that in prostate cancer screening only 30% of even the most suspicious cases identified with initial testing actually have cancer. Such class imbalance increases the FPR even higher in "critical" applications because the models tend to misclassify minority classes more often. Useful algorithms need to achieve low FPR at high TPR operating points even under class imbalance.

While various methods for learning with imbalanced datasets exist, to the best of our knowledge, these methods do not take into account the fact that "critical" applications need to be operated at high accuracy for the critical classes. We believe that for such applications ensuring low misclassification rate for the non-critical samples and high accuracy for the critical classes should be the main goal to make binary classifiers useful in practice. This motivates us to design new strategies for training DNNs for classification.

Contribution: In this paper, we pose the training of a DNN for binary classification under class imbalance as a constrained optimization problem and propose a novel constraint that can be used with existing loss functions. We define the constraint using Mann-Whitney statistics [16] in order to maximize the AUC, but in an asymmetric way to favor reduction of false positives at high true positive (or low false negative) rates. Then, we transfer the constrained problem to its dual unconstrained optimization problem using an Augmented Lagrangian method (ALM) [2]. We optimize the resulting loss function using stochastic gradient descent. Unlike the existing methods that directly optimizes AUC, we incorporate AUC optimization in a principled way into a constrained optimization framework. We finally present two possible extensions of the proposed constraint for multi-class classification problems.

We present an extensive evaluation of the proposed method for image-based binary and multi-class classification problems on three datasets: an in-house medical dataset for prostate cancer, CIFAR10, and CIFAR100 [11]. In all datasets, we performed experiments by simulating different class imbalance ratios. In our experiments, we applied the proposed constraint to 9 different baseline loss functions, most of which were proposed to handle class imbalance. We compare the results with the baselines without any constraint. The results demonstrate that the proposed method improves the baselines in majority of the cases.

2 Related work

Various methods have already been proposed to learn better models with class imbalanced datasets. We group the existing methods into three classes: cost sensitive training-based methods, sampling-based and classifier-based methods. Here, we focus on the first one and present related work for the other categories in the supplementary material for space reasons.

Cost sensitive training-based methods: This family of methods aim handling class imbalance by designing an appropriate loss function to be used during training. In particular, they design the loss functions to give more emphasis to the minority class samples, or the class with higher associated risk, than the majority ones during training [22]. [25] propose a loss function, which we refer to as Weighted BCE (W-BCE), where minority class samples are multiplied by a constant to introduce more cost to misclassification of those samples. [10] propose a method function that aims to learn more discriminative latent representations by enforcing DNNs to maintain inter-cluster and inter-class margins where clusters are formed using k-means clustering. They demonstrate that the tighter constraint inherently reduces class imbalance. In a more recent work, [5] propose a loss function called class-balanced binary cross-entropy (CB-BCE) to weight BCE inversely proportionally to the class frequencies to amplify the loss for the minority class samples. In a similar vein, [13] modifies BCE and propose symmetric focal loss (S-FL) by multiplying it with the inverse of the prediction probability to introduce more cost to the samples that DNNs are not very confident. [15] introduces symmetric margin loss (S-ML) by introducing a margin to the BCE loss. [12] investigate different loss functions such as S-FL and S-ML, and propose their asymmetric versions, A-FL and A-ML, by

introducing a margin for the minority class samples to handle class imbalance. In a different line of work, [23] propose a method called mean squared false error by performing simple yet effective modification to the mean squared error (MSE) loss. Unlike MSE, which computes an average error from all samples without considering their classes, this loss computes a mean error for each sample and averages them. [3] proposes a label-distribution-aware margin (LDAM) loss motivated by minimizing a margin-based generalization bound, optionally coupled with a training schedule that defers re-weighting until after the initial stage. [19] introduces balanced meta-softmax for long-tailed recognition, which accommodates the label distribution shift between training and testing, as well as a meta sampler that learns to re-sample training set by meta-learning. [21] proposes a loss that ignores the gradient from samples of large classes for the rare ones, making the training more fair.

A particular group within the cost sensitive training-based methods focus on optimizing AUC and our method falls into this group. AUC optimization is an ideal choice for class imbalance since AUC is not sensitive to class distributions [4]. [18] proposes a support vector machine (SVM) based loss function that maximizes AUC and demonstrates its effectiveness for the class imbalance. [26] approaches the class imbalance problem from online learning perspective and proposes an AUC optimization-based loss function. [7] proposes a one-pass method for AUC optimization that does not require storing data unlike the previous online methods. Another online AUC optimization method proposed by [24] formulates AUC optimization as a convex-concave saddle point problem. Despite their usefulness, all aforementioned AUC optimization-based methods were applied to linear predictive models and their performance on DNNs are unknown. [20] applies online AUC optimization on a small dataset for breast cancer detection where they also mention that extension to larger datasets may not be feasible. In a very recent work called mini-batch AUC (MBAUC) [8], authors extend AUC optimization to non-linear models with DNNs by optimizing AUC with minibatches and demonstrate its effectiveness on various datasets.

The proposed constrained optimization method differs from the existing works in that it enforces maximal AUC as a constraint in a way that favors reducing FPR at high TPR and can be used with existing loss functions.

3 Background - Augmented Lagrangian method (ALM)

A generic optimization problem for an objective function $F(\theta)$ subject to the constraints $\mathcal{C}(\theta) = \{c_1(\theta), ..., c_m(\theta)\}$ can be expressed as [2, 17]:

$$\arg\min_{\theta \in \Theta} F(\theta); \quad \text{subject to} \;\; \mathcal{C}(\theta) \tag{1}$$

Augmented Lagrangian method (ALM) [1], also known as methods of multipliers, converts the constrained optimization problem in Eq. (1) to an unconstrained optimization problem. ALM is proposed to overcome the limitations of two earlier methods called quadratic penalty method and method of Lagrangian multipliers which suffer from training instability and convergence due to the difficulty of convexifying loss functions[2]. In ALM, the penalty concept is merged with the primal-dual philosophy of classic Lagrangian function. In such methods, the penalty term is added not to the objective function $F(\theta)$ but rather to its Lagrangian function, thus forming the Augmented Lagrangian Function:

$$\mathcal{L}_\mu(\theta, \lambda) = F(\theta) + \mu \sum_{i=1}^{m} \left\| c_i(\theta) \right\|^2 + \sum_{i=1}^{m} \lambda_i c_i(\theta) \tag{2}$$

In practice, this method consists of iteratively solving a sequence of problems as:

$$\max_{\lambda^k} \min_{\theta} \mathcal{L}_{\mu^k}(\theta, \lambda^k), \qquad \theta \in \Theta \tag{3}$$

Where $\{\lambda^k\}$ is a bounded sequence in $\mathcal{R}^m$, updated as $\lambda_i^{k+1} = \lambda_i^k + \mu c_i(\theta)$. $\{\mu^k\}$ is a positive penalty parameter sequence, with $0 < \mu^k \leq \mu^{k+1}$, $\mu^k \to \infty$, which may be either pre-selected or generated during the computation according to a defined scheme. In ALM, increasing μ^k to indefinitely is not necessary as in the quadratic penalty method. Thus, it does not suffer from training instabilities due to the constraint prevailing $F(\theta)$. Furthermore, it does not require convexity assumption as in the method of Lagrange multipliers to ensure convergence [1].

[2]Please consult to supplementary material for more details of quadratic penalty method and method of Lagrangian multipliers.

4 Proposed methods

4.1 Proposed constraint for binary classification

Let $F(\theta)$ be a generic loss function that is used to train classification DNNs, $f_\theta(.)$, for binary problems. Let us also define $p \triangleq \{x_1^p, \cdots, x_{|p|}^p\}$ and $n \triangleq \{x_1^n, \cdots, x_{|n|}^n\}$ as the sets of positive (critical) and negative class samples, respectively. Note that we choose p as the minority class in our description, i.e., $|p| < |n|$, and assume that this is a critical class associated with higher risk of making a mistake. We define our constrained optimization problem as follows:

$$\begin{aligned} &\arg\min_{\theta} \; F(\theta) \\ &\text{subject to} \sum_{k=1}^{|n|} \max\left(0, -\left(f_\theta(x_j^p) - f_\theta(x_k^n)\right) + \delta\right) = 0, \;\; j \in \{1, ..., |p|\}, \end{aligned} \tag{4}$$

where $f_\theta(x)$ indicates output probability of the DNN on input x. Note that the constraint states that the output of the DNN for each critical class sample should be larger than the outputs of all of the negative samples by a margin δ. Satisfying the constraint would directly ensure maximal AUC.

We define the equivalent unconstrained version of Eq. (4) by writing it in the form given in Eq. (2)

$$\mathcal{L}_\mu(\theta, \lambda) = \; F(\theta) + \frac{\mu \sum_{j=1}^{|p|} q_j^2}{2 \cdot |p| \cdot |n|} + \frac{\sum_{j=1}^{|p|} \lambda_j \cdot q_j}{|p| \cdot |n|} \tag{5}$$

where, $q_j = \sum_{k=1}^{|n|} \max(0, -(f_\theta(x_j^p) - f_\theta(x_k^n)) + \delta)$, μ is the penalty coefficient corresponding to the quadratic penalty term, λ_j is the estimate of Lagrange multiplier corresponding to each positive training sample j, and δ is the margin that we determine using a validation dataset. We present the algorithm that we use during training in the supplementary material.

The crucial aspect of the above formulation is the asymmetry between positive and negative classes. A constraint is defined for each positive class sample, thus each positive class sample gets an separate Lagrange multiplier. This form prioritizes reduction of FPR at high TPR values as illustrated next.

4.2 Toy example describing our design choice

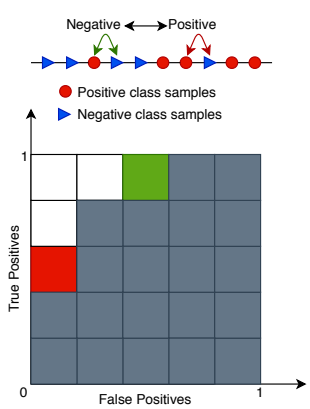

Figure 1: Toy example that illustrates different optimizations that yield the same improvement in AUC. The one that adds the green box to the AUC however, leads to lower FPRe at the highest TPRe. The ALM given in Eq. (5) prefers adding the green box to improve the AUC rather than the red one.

Recall that our goal with the constraint and the final augmented loss function is to maximize AUC through minimizing FPR for high TPR. The design of the constraint is crucial to achieve this goal. In Fig. 1, we demonstrate this on a toy example. At the top we show 10 data samples and order them with respect to a classifier's output for the samples, i.e., samples on the right are assumed to yield higher output than those on the left. The gray area in the figure below shows the AUC for the toy data samples. Consider two different optimizations to increase the AUC. One adds the red box and the other adds the green box to the gray area. Both optimizations lead to exactly the same AUC improvement, however, only adding the green box reduces FPR at the highest TPR.

In the proposed method, we design the constraint to achieve lower FPR at high TPR, such that the optimization can reduce cost more by adding the green box instead of the red box. To see this, let us assume that the distances between all the successive markers in the figure are the same, and we denote this by Δ. Note that the distances between markers indicate the differences

between the outputs of the classifiers for the samples corresponding to the markers. Let us also further assume that all the Lagrange multipliers have the same value. In this case, one can verify that if the optimization swaps the locations of the left most positive sample (red circle marker) and the negative sample to its immediate right (blue triangle marker), the cost due to the constraint in Eq. (5) will decrease by $39\mu\Delta^2/2 + 3\lambda\Delta$. Swapping the locations of the right most negative sample and the positive sample to its left will decrease the cost due to the constraint by $19\mu\Delta^2/2 + 3\lambda\Delta$[3]. So, from the cost perspective, the optimization should prefer the former swap over the latter, which corresponds to adding the green box to the AUC instead of red box. Therefore, the augmented Lagrangian cost would be decreased further when FPR at the highest TPR is reduced rather than increasing the TPR at the lowest FPR. Instead of defining the constraint for each positive sample in Eq. (5), if we were to define it for each negative sample in the exactly opposite way, i.e., $\sum_{j=1}^{|p|} \max(0, -(f_\theta(x_j^p) - f_\theta(x_k^n)) + \delta) = 0,\ k \in \{1, \ldots, |n|\}$, then the situation would have been reversed. The optimization would prefer adding the red box over the green box to decrease the cost further. If we were to define a constraint for each positive-negative pair, i.e., $f_\theta(x_j^p) > f_\theta(x_k^n), \forall j$ and k, then adding the red or the green box to improve the AUC would have yielded exactly the same decrease in the cost.

4.3 Extensions to multi-class classification

The proposed constrained can also be extended to multi-class classification with slight modifications. Let us assume that we have a multi-class classification problem with C classes. In this case, there can be multiple critical and non-critical classes. Let us define the corresponding family of sets, i.e., sets of sets, as $P = \{p^1, \cdots, p^{|P|}\}$ and $N = \{n^1, \cdots, n^{|N|}\}$ where the sets p^i and n^i are as defined in the previous section. Also, note that $|P \cup N| = C$ and $P \cap N = \emptyset$.

A main difference between binary and multi-class classification is the dimension of the output. While $f_\theta(x)$ was a single value for binary classification, in multi-class problem it is a vector with one value for each class. Using the notation for the positive and negative classes, we write the output of the network as $f_\theta(x) = \{f_\theta^{p^1}, \cdots, f_\theta^{p^{|P|}}, f_\theta^{n^1}, \cdots, f_\theta^{n^{|N|}}\}$. Based on this, we define our first constraint for multi-class classification as

$$q_{cj} = \sum_{i=1}^{|N|} \sum_{k=1}^{|n^i|} \max\left(0, -\left(f_\theta^{p^c}(x_j^{p^c}) - f_\theta^{p^c}(x_k^{n^i})\right) + \delta\right), \quad c \in \{1, \cdots, |P|\},\ j \in \{1, ..., p^c\} \quad (6)$$

where $f_\theta^{p^c}$(x) indicates the output probability of DNN at the critical class p^c. Note that the constraint in Eq. (6) enforces that the output probability of the DNN for critical classes should be larger for critical class samples than that of other classes.

An alternative definition of the constraint is also possible. The first constraint enforces that critical class probabilities should be larger for critical class samples. We can go a step further and add penalty to enforce that non-critical class probabilities should be smaller for critical samples than the non-critical samples belonging to that class.

$$\begin{aligned} q_{cj} = \sum_{i=1}^{|N|} \sum_{k=1}^{|n^i|} & \max\left(0, -\left(f_\theta^{p^c}(x_j^{p^c}) - f_\theta^{p^c}(x_k^{n^i})\right) + \delta\right) \\ & + \max\left(0, \left(f_\theta^{n^i}(x_j^{p^c}) - f_\theta^{n^i}(x_k^{n^i})\right) + \delta\right), \quad c \in \{1, \cdots, |P|\},\ j \in \{1, ..., p^c\} \end{aligned} \quad (7)$$

In this formulation, the penalty does not view all non-critical classes as one but also contributes to increasing classification accuracy in those classes as well by increasing the gap between the corresponding probability of critical class and non-critical class samples.

5 Experiments

In this section, we present our experimental evaluations on image-based classification tasks. We perform experiments on three datasets: an in-house MRI medical dataset for prostate cancer and two publicly available computer vision dataset, CIFAR10 and CIFAR100 [11].

[3] Derivations for this toy example and theoretical insight for our design choice are provided in the supplementary material for convenience.

In our evaluation, we experiment with different existing loss functions most of which have been designed to handle class imbalance: classic binary and multi-class cross-entropy (BCE, CE), symmetric margin loss (S-ML) [15], symmetric focal loss (S-FL) [13], asymmetric margin loss (A-ML) and focal loss (A-FL) [12], cost-weighted BCE (WBCE) [25], class-balanced BCE (CB-BCE) and CE (CB-CE) [5], label-distribution-aware margin loss (LDAM) [3]. We first train DNNs for classification using only the loss functions and then using our method, which adds the proposed constraint to the loss function and solves Eq. (5), and we compare the classification performances. In addition, we also compare the proposed method with directly optimizing AUC using the mini-batch AUC (MBAUC) method proposed in [8] for the binary case.

The proposed method is implemented in PyTorch and we run all experiments on a Nvidia GeForce GTX Titan X GPU with 12GB memory.

5.1 Datasets

Prostate MRI dataset consists of 2 distinct cohorts: 1) a group of 300 multiparametric prostate MRI studies used for training and 2) another group of 100 multiparametric prostate MRI studies used for testing the trained DNNs. There is no overlap between the groups. Consent from the each subject is obtained to use the data for research purposes. Two board-certified radiologists with 10 and 7 years of experience in dedicated prostate imaging independently reviewed all examinations of the training set and test set and scored whether Dynamic Contrast Enhanced (DCE) sequences would have been beneficial for diagnosis. After completion of readings a consensus was reached by the two readers by reviewing all examinations with discrepant decisions. The goal of the binary classification here is to identify subjects who do not require additional DCE imaging for accurate diagnosis, so they can be spared from unnecessary injection and cost and duration of the scanning can be reduced. In our experiments, we randomly split 20% of the training cohort as validation set by keeping the class imbalance consistent across the datasets. In both training and testing cohorts, positive samples represent 13% of all the patients which leads to an inherent 1:8 class ratio.

CIFAR10 and CIFAR100 datasets For CIFAR10, in the binary experiments we randomly select 2 classes among 10 to pose a binary classification problem. Additional experiments with different random selections are in the supplementary material. We use all the available training samples of the selected majority class, while we randomly pick a varying number of training samples for the minority class to obtain different class ratios, up to 1:200. We select 100 samples per class as validation set to determine hyper-parameters. For testing, we use all the available test samples, which consist of 1000 images for each class. For CIFAR100 in the binary setup we select one super-class as majority one, and a sub-class of another super-category as minority one. For training we use all the available 2250 samples of the selected larger class, and we randomly pick a varying number of training samples for the minority class to obtain different class ratios, up to 1:200. We use 50 samples for each original sub-class as validation set to perform the parameters' search. For testing, we use all the available test samples, which consist of 100 images for each sub-class. For the multi-class experiments, long-tailed versions of CIFAR10 and CIFAR100 are built accordingly to [5, 3] and we consider as critical class the smallest one.[4]

5.2 Training details

Network architectures: For the prostate MRI dataset, we use a 3D CNN that consist of cascaded 3D convolution, 3D max-pooling, intermediate ReLu activation functions and Sigmoid in the final output. For the binary CIFAR10 and CIFAR100 datasets, we use ResNet-10, while ResNet-32 [9] is adopted for the multi-class experiments, consistently with [3, 5]. Further training details can be found in the supplementary material.

Ensembling for higher reliability: Model reliability is very crucial when training DNNs. Dealing with small datasets may lead to dataset-dependent results even with the random splits, which could completely hinder objective evaluation. To weaken this phenomenon, we adopt the following ensembling strategy on MRI and for consistency we apply it on the binary CIFAR10 and CIFAR100 as well. Given a dataset and a class ratio, we create 10 random stratified splits of the dataset and train 10 models independently. The larger portions are used for training and the smaller portions for choosing hyper-parameters. During inference, all the models are applied on test samples and

[4]Experiments with multiple critical classes are presented in supplementary materials.

predictions are averaged in the logit space before the sigmoid function to yield the final prediction. We apply the ensembling to all the binary models we experiment with. This practice attenuates data dependency and we observed that the final AUC is improved when compared to the average of AUCs of different models, as presented in Table 3 for MRI sequences.

Hyper-parameters selection: Selection of the best hyper-parameters is crucial both to ensure proper and fair evaluation of the methods and to understand the true performance of any model. To achieve this, we performed grid-search to determine the hyper-parameters that yielded the highest AUC for the binary experiments. For the multi-class tests we selected the model that achieved the best overall accuracy on the validation set, in order to be consistent with the related works. The test sets in all experiments were not used for hyper-parameter selection. To reduce computational load, we selected the common hyper-parameters such as the optimizer, learning rate, and the activation functions in the DNNs based on their performance with BCE loss function for the binary experiments, and consistently with [3] for the multi-class ones. Then, we keep them fixed in all experiments on the same dataset.

Besides the common hyper-parameters, the majority of the existing methods have hyper-parameteres that crucially affect their performance. Namely, these hyper-parameters are margin m for S-ML and A-ML, exponent γ for S-FL and A-FL, weight of the cost c for WBCE and β for CB-BCE. We selected best values for these hyper-parameters from respective candidate sets that we created based on the information provided in the original papers for each of them.

In the proposed method, there are 4 parameters to be set: $\mu^{(0)}$, $\lambda^{(0)}$, ρ, and δ. Thus, hyperparameter search is an important aspect for the proposed method. $\mu^{(0)}$, $\lambda^{(0)}$, and ρ are stemming from ALM and we follow the guideline from [2] when setting them. We initialized all the Lagrangian multipliers $\lambda_i^{(0)}$ to 0. We choose $\mu^{(0)}$ from the set $\{10^{-7}, 10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}\}$, as it is suggested to choose a small value in the beginning and increase it iteratively using the equation $\mu^{(k+1)} = \rho \cdot \mu^{(k)}$. We choose ρ from the set $\{2, 3\}$ as $\rho > 1$ is suggested. Moreover, we do not increase ρ beyond 4 to avoid potential dominance of the constraint on $F(\theta)$ since ρ is used to increase μ. Once we find the best combination of μ and ρ based on the AUC on the validation set, we fix them and we search for δ as final step. Please see supplementary materials for further details on the hyper-parameters selection.

Once the hyper-parameters for each model are selected, the training is performed and models are applied to the test sets to yield the final results, which is described next.

Table 1: Results on binary CIFAR10 for class ratio 1:100 and 1:200.

Dataset	Binary CIFAR10, imb. 100				Binary CIFAR10, imb. 200			
Training method	FPR @ 98% TPR	FPR @ 95% TPR	FPR @ 92% TPR	Test AUC	FPR @ 98% TPR	FPR @ 95% TPR	FPR @ 92% TPR	Test AUC
BCE	56.0	45.0	29.0	91.2	75.0	55.0	40.0	87.3
S-ML	59.0	40.0	26.0	91.7	75.0	54.0	<u>35.0</u>	87.4
S-FL	59.0	40.0	27.0	91.7	78.0	59.0	43.0	85.7
A-ML	54.0	36.0	23.0	92.4	74.0	56.0	39.0	87.4
A-FL	50.0	38.0	24.0	92.3	76.0	59.0	40.0	86.2
CB-BCE	89.0	72.0	59.0	78.0	87.0	74.0	61.0	78.0
W-BCE	69.0	52.0	37.0	87.4	88.0	75.0	62.0	78.3
LDAM	65.0	48.0	34.0	89.0	78.0	63.0	45.0	86.4
MBAUC	86.0	71.0	56.0	74.0	89.0	83.0	69.0	67.9
ALM + BCE	**52.0**	**<u>34.0</u>**	**<u>21.0</u>**	**<u>93.1</u>**	**<u>70.0</u>**	**54.0**	**39.0**	86.7
ALM + S-ML	**50.0**	**37.0**	**24.0**	**92.5**	**72.0**	**<u>52.0</u>**	39.0	**<u>87.9</u>**
ALM + S-FL	**55.0**	**39.0**	**25.0**	91.5	**74.0**	**55.0**	**41.0**	**86.9**
ALM + A-ML	**<u>45.0</u>**	**35.0**	23.0	**92.8**	75.0	**74.0**	**<u>35.0</u>**	**87.6**
ALM + A-FL	**49.0**	**37.0**	**23.0**	**92.7**	78.0	**57.0**	**37.0**	**87.0**
ALM + CB-BCE	**67.0**	**51.0**	**36.0**	**88.1**	**85.0**	**69.0**	**53.0**	**80.0**
ALM + W-BCE	**66.0**	**48.0**	**31.0**	**89.3**	**83.0**	**69.0**	**54.0**	**81.0**
ALM + LDAM	**60.0**	**42.0**	**31.0**	**91.0**	**73.0**	**61.0**	**43.0**	85.6

Table 2: Results on binary CIFAR100 for class ratio 1:100 and 1:200.

Dataset	Binary CIFAR100, imb. 100				Binary CIFAR100, imb. 200			
Training method	FPR @ 98% TPR	FPR @ 95% TPR	FPR @ 90% TPR	Test AUC	FPR @ 98% TPR	FPR @ 95% TPR	FPR @ 90% TPR	Test AUC
BCE	93.0	63.0	47.0	81.8	94.0	77.0	61.0	79.1
S-ML	89.0	65.0	43.0	82.7	95.0	75.0	64.0	79.7
S-FL	89.0	62.0	44.0	82.6	90.0	78.0	50.0	80.1
A-ML	91.0	63.0	44.0	81.8	95.0	75.0	66.0	79.8
A-FL	88.0	63.0	45.0	82.8	91.0	78.0	50.0	80.0
CB-BCE	93.0	75.0	52.0	78.8	93.0	78.0	51.0	78.7
W-BCE	88.0	59.0	41.0	79.7	95.0	63.0	51.0	79.7
LDAM	84.0	70.0	42.0	82.8	80.0	67.0	45.0	82.1
MBAUC	81.0	62.0	41.0	82.3	88.0	63.0	48.0	80.3
ALM + BCE	**91.0**	**49.0**	**39.0**	**82.7**	**87.0**	**66.0**	**57.0**	**80.9**
ALM + S-ML	**88.0**	69.0	**41.0**	81.7	**87.0**	**73.0**	**55.0**	**80.7**
ALM + S-FL	**88.0**	**60.0**	**42.0**	81.7	**85.0**	**76.0**	50.0	**80.8**
ALM + A-ML	**89.0**	**55.0**	**37.0**	**82.7**	**92.0**	**63.0**	**45.0**	**81.0**
ALM + A-FL	**86.0**	**62.0**	**40.0**	**83.2**	**88.0**	**76.0**	**46.0**	**80.7**
ALM + CB-BCE	**89.0**	**59.0**	**36.0**	**83.8**	**85.0**	**66.0**	**44.0**	**81.0**
ALM + W-BCE	**87.0**	**53.0**	**39.0**	**83.2**	**79.0**	**62.0**	**44.0**	**81.3**
ALM + LDAM	**80.0**	**59.0**	**40.0**	**83.2**	84.0	**61.0**	46.0	81.5

5.3 Results

We present binary classification results on CIFAR10, CIFAR100, and the in-house medical imaging datasets in Tables 1, 2 and 3, respectively. We evaluate the performance of the baselines and the proposed method (ALM) using FPR at maximal levels of TPR (or minimal levels of false negative rate (FNR)) and AUC on the test sets. In the binary experiments, ALM is overall able to consistently improve the performance of almost all the loss functions with regard to both AUC and FPRs at maximal TPR levels. Even in those cases when AUC is improved to a moderate extent there is still an improvement in FPRs, in accordance with our goal. Moreover, it is noticeable that the higher the TPR, the higher the benefit of applying ALM, which is in accordance with our target applications. In fact, considering that such classifiers in "critical" applications would be operated at high TPR (or low FNR), reduction in FPR in these settings is the effect we desired from the proposed approach. Lastly, we also observe in the tables that directly optimizing AUC via MBAUC does not provide the same improvements as using the proposed ALM approach. We present additional binary classification experiments in the supplementary material.

Additionally, we present the quantitative results of multi-class experiments in Table 4. Comparison with the other baselines are presented in the supplementary materials. In these experiments, we evaluate the performance by computing the accuracy on the non-critical classes, at various levels of TPR for the critical-class. For this purpose, the first step is to find a threshold on the critical class' logit such that the desired TPR is obtained for the important class. All the test samples whose critical class' logit exceeds the selected threshold are assigned to the important class. The remaining samples that are not assigned to the critical class are then classified based on the highest probability over the non-critical logits. In addition to this metric, we present overall classification accuracy on all classes. The quantitative results demonstrate that both of the multi-class strategies, $\text{ALM}_{m,1}$ and $\text{ALM}_{m,2}$ improve on the baseline, reducing the error on non-critical classes, at high levels of accuracy on the important class. Moreover, $\text{ALM}_{m,2}$ is able to further improve the overall accuracy by a larger margin in almost all the experiments thanks to the additional term in the constraint.

6 Conclusion

In this paper, we pose the training of a DNN for binary classification under class imbalance as a constrained optimization problem and propose a novel constraint that can be used with existing loss

Table 3: Results on in-house MRI dataset.

Method	FPR @0 FN	FPR @1 FN	Avg AUC	AUC ens.
BCE	80.0	80.0	65.4±9.0	70.9
S-ML	81.0	77.0	67.3±7.0	71.5
S-FL	77.0	38.0	71.7±10.0	80.3
A-ML	77.0	73.0	68.0±9.0	74.2
A-FL	66.0	38.0	67.7±8.0	80.1
CB-BCE	100.0	34.0	72.0±5.0	77.7
W-BCE	56.0	42.0	68.8±6.0	80.5
LDAM	100.0	75.0	62.0±9.0	66.4
MBAUC	61.2	33.0	71.2±11.0	82.4
ALM + BCE	**54.0**	**38.0**	76.8±9.0	**85.4**
ALM + S-ML	**81.0**	**33.0**	72.5±9.0	**80.3**
ALM + S-FL	**53.0**	**26.0**	72.5±10.0	**84.2**
ALM + A-ML	**72.0**	**53.0**	67.2±5.0	**76.4**
ALM + A-FL	**62.0**	46.0	74.7±7.0	**81.5**
ALM + CB-BCE	**86.0**	34.0	73.0±9.0	**79.5**
ALM + W-BCE	59.0	**40.0**	72.4±6.0	**81.4**
ALM + LDAM	**59.0**	**53.0**	66.5±8.5	**77.0**

Table 4: Results on long-tailed CIFAR10 for class imbalance 1:100 and 1:200. The Table shows the error on all the non-important classes, after setting a threshold on the important class' logit to obtain 80, 90% TPR.

Dataset	Long-tailed CIFAR10, imb. 100			Long-tailed CIFAR10, imb. 200		
Training method	Error @ 80% TPR	Error @ 90% TPR	Overall Accuracy	Error @ 80% TPR	Error @ 90% TPR	Overall Accuracy
CE	29.80	34.67	70.35	37.81	42.37	64.03
FL	32.21	36.47	69.20	38.50	42.21	62.90
CB-BCE	31.04	33.44	72.90	35.11	39.13	65.77
LDAM	26.57	29.86	71.80	34.41	45.72	65.87
$\text{ALM}_{m,1}$ + CE	**28.89**	**33.93**	**70.90**	**36.14**	**39.90**	**65.13**
$\text{ALM}_{m,1}$ + FL	**29.59**	**34.94**	**69.74**	**36.87**	**41.87**	**64.23**
$\text{ALM}_{m,1}$ + CB-CE	**27.89**	**30.27**	72.10	**33.09**	**35.44**	65.34
$\text{ALM}_{m,1}$+ LDAM	**25.73**	**28.52**	**72.86**	**31.94**	**37.41**	65.65
$\text{ALM}_{m,2}$ + CE	**29.53**	**34.09**	**71.30**	**35.10**	**39.19**	**64.35**
$\text{ALM}_{m,2}$ + FL	**30.50**	**35.63**	**69.47**	**36.27**	**40.43**	**64.43**
$\text{ALM}_{m,2}$ + CB-CE	**27.84**	**31.97**	72.09	**33.72**	**36.86**	**66.04**
$\text{ALM}_{m,2}$ + LDAM	**24.76**	**28.91**	**73.32**	**31.02**	**36.09**	**67.41**

functions. The proposed constraint is designed to maximize the AUC, but in an asymmetric way to favor reduction of FPR at high TPR (or low FNR). Then, we transfer the constrained problem to its dual unconstrained optimization problem using an Augmented Lagrangian method (ALM) [2] which we optimize using stochastic gradient descent. Additionally, we presented two possible extensions of the proposed constraint for multi-class classification problems.

We performed an extensive evaluation of the proposed constraints for binary and multi-class image classification problems on both computer vision and medical imaging datasets. We compare the performance of the proposed constraints with different baselines by simulating different class imbalance ratio. The quantitative results demonstrate that the proposed constraints improves the performance of the baselines in majority of the cases in both binary and multi-class classification experiments.

References

[1] Bertsekas, D.P.: Multiplier methods: A survey. Automatica **12**(2), 133 – 145 (1976). https://doi.org/https://doi.org/10.1016/0005-1098(76)90077-7, http://www.sciencedirect.com/science/article/pii/0005109876900777

[2] Bertsekas, D.: Nonlinear Programming. Athena Scientific (1999)

[3] Cao, K., Wei, C., Gaidon, A., Arechiga, N., Ma, T.: Learning imbalanced datasets with label-distribution-aware margin loss. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 32. Curran Associates, Inc. (2019), https://proceedings.neurips.cc/paper/2019/file/621461af90cadfdaf0e8d4cc25129f91-Paper.pdf

[4] Cortes, C., Mohri, M.: Auc optimization vs. error rate minimization. Advances in neural information processing systems **16**(16), 313–320 (2004)

[5] Cui, Y., Jia, M., Lin, T.Y., Song, Y., Belongie, S.: Class-balanced loss based on effective number of samples. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9268–9277 (2019)

[6] Djavan, B., Zlotta, A., Remzi, M., Ghawidel, K., Basharkhah, A., Schulman, C., Marberger, M.: Optimal predictors of prostate cancer on repeat prostate biopsy: a prospective study of 1,051 men. The Journal of urology **163 4**, 1144–8; discussion 1148–9 (2000)

[7] Gao, W., Jin, R., Zhu, S., Zhou, Z.H.: One-pass auc optimization. In: International conference on machine learning. pp. 906–914. PMLR (2013)

[8] Gultekin, S., Saha, A., Ratnaparkhi, A., Paisley, J.: Mba: mini-batch auc optimization. IEEE transactions on neural networks and learning systems **31**(12), 5561–5574 (2020)

[9] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. pp. 770–778 (06 2016). https://doi.org/10.1109/CVPR.2016.90

[10] Huang, C., Li, Y., Loy, C.C., Tang, X.: Learning deep representation for imbalanced classification. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5375–5384 (2016)

[11] Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)

[12] Li, Z., Kamnitsas, K., Glocker, B.: Overfitting of neural nets under class imbalance: Analysis and improvements for segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. pp. 402–410. Springer (2019)

[13] Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision. pp. 2980–2988 (2017)

[14] Litjens, G., Kooi, T., Bejnordi, B.E., Setio, A.A.A., Ciompi, F., Ghafoorian, M., van der Laak, J.A., van Ginneken, B., Sánchez, C.I.: A survey on deep learning in medical image analysis. Medical Image Analysis **42**, 60–88 (Dec 2017). https://doi.org/10.1016/j.media.2017.07.005, http://dx.doi.org/10.1016/j.media.2017.07.005

[15] Liu, W., Wen, Y., Yu, Z., Yang, M.: Large-margin softmax loss for convolutional neural networks. In: ICML. vol. 2, p. 7 (2016)

[16] Mann, H.B., Whitney, D.R.: On a test of whether one of two random variables is stochastically larger than the other. The annals of mathematical statistics pp. 50–60 (1947)

[17] Nocedal, J., Wright, S.J.: Numerical Optimization. Springer, New York, NY, USA, second edn. (2006)

[18] Rakotomamonjy, A.: Optimizing area under roc curve with svms. In: ROCAI. pp. 71–80 (2004)

[19] Ren, J., Yu, C., Sheng, S., Ma, X., Zhao, H., Yi, S., Li, H.: Balanced meta-softmax for long-tailed visual recognition. In: Proceedings of Neural Information Processing Systems(NeurIPS) (Dec 2020)

[20] Sulam, J., Ben-Ari, R., Kisilev, P.: Maximizing auc with deep learning for classification of imbalanced mammogram datasets. In: VCBM. pp. 131–135 (2017)

[21] Tan, J., Wang, C., Li, B., Li, Q., Ouyang, W., Yin, C., Yan, J.: Equalization loss for long-tailed object recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2020)

[22] Thai-Nghe, N., Gantner, Z., Schmidt-Thieme, L.: Cost-sensitive learning methods for imbalanced data. In: The 2010 International joint conference on neural networks (IJCNN). pp. 1–8. IEEE (2010)

[23] Wang, S., Liu, W., Wu, J., Cao, L., Meng, Q., Kennedy, P.J.: Training deep neural networks on imbalanced data sets. In: 2016 international joint conference on neural networks (IJCNN). pp. 4368–4374. IEEE (2016)

[24] Ying, Y., Wen, L., Lyu, S.: Stochastic online auc maximization. In: NIPS. pp. 451–459 (2016)

[25] Zadrozny, B., Langford, J., Abe, N.: Cost-sensitive learning by cost-proportionate example weighting. In: Third IEEE international conference on data mining. pp. 435–442. IEEE (2003)

[26] Zhao, P., Hoi, S.C., Jin, R., YANG, T.: Online auc maximization (2011)

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] Please, refer to Section 5.2.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] The proposed method aims to improve the performance on critical, under-represented classes. Fairness is one of the motivations of this work.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The code is provided in the supplementary material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Training details are provided in Section 5.2 and in supplementary material.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] Confidence intervals for the proprietary MRI dataset are reported in Table 3. For the binary CIFAR10 and CIFAR100 they are reported in the supplementary materials.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] The adopted computational resources are described in Section 5.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] The code for implementing the proposed method is provided as supplementary material
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] MRI dataset creation is described in Section 5.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A] We did not use crowdsourcing for this work.
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]