Reproduction of Baselines on Label-Distribution-Aware Margin Loss and Deferred Reweighting Schedule

Xinyu He, Hehuimin Cheng, Vitaly Kondulukov
Department of Computer Science
McGill University

December 26, 2019

Abstract

Most state-of-the-arts classifiers assume a relatively balanced class distribution and equal misclassification cost. Training with imbalanced data has encountered a significant difficulty of low attainable results. Although many previous work has addressed various strategies to tackle this issue, these techniques usually come with different drawbacks and the outcome is still very limited. Cao et al. introduced two new techniques, label-distribution-aware margin loss (LDAM) and deferred re-weighting(DRM) [1], which have been claimed to acieve better performance gains over the existing techniques. In this work, we reproduced the baseline experiments reported in the authors' work with IMDB and CIFAR-10 benchmarks. We performed extensive hyper-parameter tuning on these models and outperformed the original reported results. We also proposed a general scheme for baseline improvement with learning rate step decay and triangular policy[2]. Based on the improved results, we studied how different techniques affect the performance when learning imbalanced data (Section 6.3.4), including class balanced re-weighting[3], class balanced re-sampling[3] and borderline-SMOTE[4].

1 Introduction

Real world data commonly show the particularity to have a number of samples of a given class under represented compared to other classes. Most existing machine learning algorithms optimize the overall accuracy without taking into account the relative distribution of each class. Without good strategies to overcome the class imbalance problem, most classifiers perform very poorly on the minority classes. Recent break-through work, *Learning Imbalanced Datasets with Label-Distribution-Aware Margin Loss*, is published by Cao et al. in NeurIPS 2019 [1]. The main contributions of this work are summarised as follows:

- A theoretically-principled new loss function: Label-Distribution-Aware Margin loss (LDAM)
- A training process: deferred re-weighting (DRW) or deferred re-sampling (DRS) optimizing schedule
- A combined optimal solution: deferred re-weighting training schedule with label-distribution-aware margin loss(LDAM + DRW)

In this work, we reproduced and inspected a subset of the baseline experiments in Cao et al.'s work and extended the baseline experiments with SMOTE oversampling experiment. Our work consists of two groups of experiments: one groups was based on IMDB movie reviews and another one was based on CIFAR-10. With the same hyper-parameters, we produced similar results as reported in the original paper. We analyzed the learning behaviour for the baselines and proposed a scheme for hyper-parameter tuning to improve the baseline results. For IMDB experiments, the fine tuning of bidirectional LSTM was able to improve around 1% compared to the top reported result. We also investigated in the performance of traditional linear models, such as Logistic Regression, Naive Bayes, etc. For CIFAR-10 experiments, our method of learning rate step decay with triangular policy in the last stage of training was able to increase the accuracy on all the baseline models. With stronger model performance, we performed detailed studies on these baseline models to understand how different techniques affect

the performance when learning imbalanced data. Finally, we incorporate our scheduling method with deferred re-weighting (DRW). We demonstrate a process of finding the optimal DRW or DRS stage transition point. With the same experiment setting, our procedure was able to improve around 2% compared to the reported results.

2 Background

Our main focus in this work is the proposed methods in *Learning Imbalanced Datasets with Label-Distribution-Aware Margin Loss* by Cao et al.[1]. Here are the brief introductions of their methods.

2.1 Label-Distribution-Aware (LDAM) Margin Loss

LDAM loss is a class-dependent soft margin loss function inspired by multi-class extension of hinge loss and cross entropy loss. The authors suggest that the non-smoothness of hinge loss may pose difficulties for optimization. Therefore, they presented a smooth relaxation of the hinge loss. It is essentially the cross entropy loss with enforced margins depending on the class distribution. LDAM loss is defined with the hyper-parameter C as: Let the training margin for class j be:

$$\gamma_j = \frac{C}{(n_j)^{1/4}} \text{ for } j \in 1, ..., k \tag{1}$$

Let (x, y) be an example and f be a model.
Let $z_j = f(x)_j$ be j-th output of the model for the j-th class. The LDAM loss is defined as:

$$L_{LDAM}((x, y); f) = -log\frac{e^{x_y - \Delta_y}}{e^{x_y - \Delta_y} + \Sigma_{j \neq y} e^{z_j})}$$
$$\text{where } \Delta_j = \frac{C}{(n_j)^{1/4}} \text{ for } j \in 1, ..., k \tag{2}$$

It is evident that LDAM loss function encourages larger margins on minority classes and smaller margins on majority classes.

2.2 Deferred Re-weighting (DRW) or Re-sampling Optimizing Schedule(DRS)

DRW/DRS schedule has two stages of training. The first stage of training uses vanilla empirical risk minimization (ERM). It is followed by the second stage of training with annealed learning rate and re-weighting or re-sampling techniques. The DRW/DRS schedule is proposed since the authors discovered the features produced by re-weighting and re-sampling before annealing the learning rate are worse than those produced by vanilla ERM.

2.3 Combined solution (LDAM + DRW)

The authors claimed that either LDAM or DRW performs over the existing techniques; however, the combination algorithm(shown in Figure8) was able to achieve the best result among all the experiments.

3 Related work

To handle the difficulties of learning imbalanced datasets, the existing solutions can be divided into data level and algorithmic level. At data level, there are some strategies to reduce the skewness of class distribution prior to the training process. As a result, the algorithms are more likely to detect the minority during the training process instead of treating the minor classes as noises. At algorithmic level, various modern techniques are proposed in the last decade targeting the optimization of training imbalanced data. The LDAM-DRW approach is a combination of data level and algorithmic level approaches that achieves a better performance gain over the existing techniques.

3.1 Re-sampling and SMOTE:

Batuwita and Palade[5] discussed in details that oversampling can produce better classification results than undersampling by increasing the minority class recognition rate without losing much of the majority class recongnition rate. However, oversampling can suffer greatly from over-fitting the minority classes and degrade the learning quality. N. V. Chawla et al.[6] proposed SMOTE method to generate new synthetic examples by interpolating between the minority examples and their selected nearest neighbours. Based on the traditional SMOTE method, H. Han et al.[4] proposed two new methods, borderline-SMOTE1 and borderline-SMOTE2. The intuition of borderline-SMOTE is illustrated in Figure1. Since the borderline minority class examples are more easily to

be misclassified than the ones far from the borderline. Thus oversampling only the borderline examples instead of all examples greatly reduces the overfitting problem.

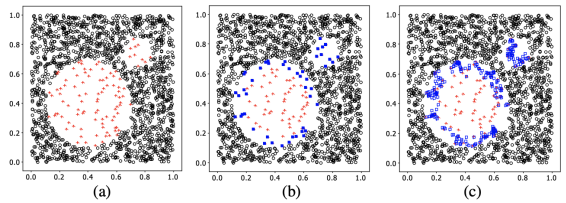

Figure 1: (a) The original distribution; (b) Borderline minority examples (blue solid squares); (c) Borderline synthetic minority examples. Figures taken from the paper *Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning*[4]

3.2 Re-weighting:

The imbalance of labels usually brings the distribution mismatch, leading to the overfitting to bias and label noise. In addition to re-sampling, reweighting is another approach researchers often use to solve imbalance of dataset[7]. Compared to the standard objective function $\frac{1}{N}\sum_{i=1}^{N} f_i(\theta)$, where $f_i(\theta)$ to represent the loss function associating with data, we introduced training hyper parameters w^* into the objective function, which can be rewrite as:

$$\theta^*(w) = \arg\min_{\theta} \sum_{i=1}^{N} w_i f_i(\theta) \qquad (3)$$

The collection of w can be selected based on the validation performance:

$$w^* = \arg\min_{w, w\geq 0} \frac{1}{M} \sum_{i=1}^{M} f_i^v(\theta^*(w)) \qquad (4)$$

w_i should larger or equal to 0 for all i, since the minimization of the negative training loss can result in instability of model.

3.3 Class-Balanced Loss Based on Effective Number of Samples

Cui et al.[3] argue that as number of samples increases, the additional benefit of a newly added data point decreases. They quantify the effective number of samples in a large-scaled dataset by a single formula, where n is the number of samples:

$$(1-\beta^n)/(1-\beta), \text{ where } \beta \in [0, 1) \qquad (5)$$

By adding this class-balanced term to existing to existing commonly used loss functions, such as softmax cross-entropy, sigmoid cross-entropy and focal loss, can achieve a significant performance improvement.

3.4 Step Decay and Cyclical Learning

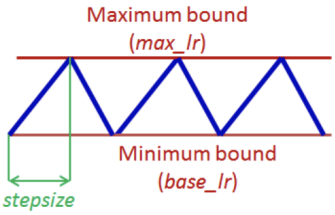

Figure 2: Triangular learning rate policy: The blue lines represent learning rate values changing between bounds. Step size is the number of iterations in a half cycle.

When training a deep neural network, adaptive learning rate usually settles down into deeper and narrower local minima of the loss function [8]. In practice, step decay is often used since it is easier for hyper-parameters interpretation. Decaying learning rate accelerates the convergence of Stochastic Gradient Decent (SGD) and often provides a more stabilized learning result. In 2017, Leslie N. Smith[2] described a new method to set the learning rate, named cyclical learning learning rates. Instead of monotonically decrease the learning rate, this method lets the learning rate cyclically vary between reasonable boundaries. Dauphin et al. [9] argue that the training difficulty arises when the saddle point plateaus with small gradients. Small gradients slow down the learning process; however, cyclical learning rate allows to increase the learning rate to traverse the saddle surface more rapidly. In the paper *Cyclical Learning Rates for Training Neural Networks*, it is also mentioned that linearly increasing the learning rate for a few epochs in the beginning can help with estimating the reasonable bounds. In our work, we adapted both step decay and cyclical learning rate with DRW learning schedule. Specifically, we adopted the triangular learning rate policy at both stages of learning. Triangular policy interpolates the learning rate linearly within the

boundaries, which is illustrated in Figure2. The learning rate at each epoch can be calculated as:

$$\begin{aligned} \eta &= \eta_{min} + (\eta_{max} - \eta_{min})(\max(0, 1 - x)) \\ \text{where } x &= |\frac{epochs}{stepsize} - 2 \text{ cycle} + 1| \\ \text{where} cycle &= floor\ (\frac{1 + epochs}{2 \text{ stepsize}}) \end{aligned} \tag{6}$$

4 Dataset and Setup

In this work, we strictly re-constructed and improved the baseline experiments on IMDB and CIFAR-10 dataset that Cao et al. proposed in the paper *Learning Imbalanced Datasets with Label-Distribution-Aware Margin Loss*[1].

The IMDB includes 50,000 well-balanced movie reviews for natural language processing or text analytics. Specifically, it includes 25,000 reviews for training and 25,000 for testing. To generate the imbalanced dataset, we removed 90% of the negative reviews manually. For the cleaning, we lowered, lemmatized each token and removed the stop words as well as non-character tokens in the sentences. We used a ratio of 0.8/0.2 to split the development set.

The original version of CIFAR-10 is composed of 10 balanced classes of 50000 training images and 10000 validation images of size 32 x 32. We manually created an unbalanced training set by removing some amounts of the training examples in every class. We manually simulated a long-tailed imbalance situation with CIFAR-10 data. The sample sizes across different classes follow a continuous distribution with an exponential decay. The ratio of class sizes between the most frequent class and the least frequent class is defined as:

$$\rho = \frac{max_i(n_i)}{min_i(n_i)} \tag{7}$$

We simulated an extreme imbalanced setting, in which the majority class size is 100 times bigger than the minority class size (ρ=100). We do not modify the original balanced validation set and use it to evaluate the algorithms' performance among all classes.

5 Evaluation Metrics in Imbalanced Domains

To evaluate the performance of a model for imbalanced datasets, accuracy generally favours the majority classes more and is much less reliable for the minority classes. However, in our experiment setup, we manually simulated the imbalanced data and kept the validation set uniformly distributed; therefore, accuracy as an evaluation metric is still relatively reasonable to use. However, with the real life imbalanced data in which the validation set is also imbalanced, it is important to change the performance metric as the accuracy or the error rate can be very misleading. Some more reasonable metrics are confusion matrix, precision, recall and F1 score. In this work, all the results are reported in terms of the $ErrorRate = 100(1 - accuracy)$, in order to to be consistent with the reference paper.

6 Experiments and Results

In this section, we describe our experiments for reproduction, inspect and improve the baseline results reported in *Learning Imbalanced Datasets with Label-Distribution-Aware Margin Loss*[1]. We re-implemented the learning algorithms based on the released code by the authors `https://github.com/kaidic/LDAM-DRW`.

6.1 Summary of Reproduced Experiments

We start by reproducing a series of results including the baselines and the proposed models. In the original paper, the authors used empirical risk minimization loss (ERM) (ie. standard unweighted cross entropy loss) as a baseline to compare the existing techniques and their proposed methods. The author applied several state-of-the-art techniques to mitigate the issues with training on imbalanced data. We investigated a portion of the baselines from the paper and also designed an additional baseline with SMOTE oversampling. Here is a list of our experimented baselines:

- empirical risk minimization with cross entropy loss (ERM)
- SMOTE Re-weighting (SMOTE)
- Class Balanced Re-weighting (CB-RWB)

- Class Balanced Re-sampling (CB-RS)

Finally we validate the authors' techniques:

- Label-Distribution-Aware Margin Loss (LDAM)
- Deferred Re-weighting Schedule (DRW)
- Combination of the two methods (LDAM+DRW)

Our investigations on the baselines consists of two groups: a text classification task on IMDB dataset and an image classification task on CIFAR-10 dataset. Our baseline investigation summaries for IMDB and CIFAR-10 are reported in Table1 and Table2 respectively.

Experiment	ErrorP	ErrorN	ErrorM
ERM	**6.32**	**26.45**	**16.48**
RS	7.90	31.93	19.916
RW	7.14	31.24	19.19
SMOTE	14.74	27.94	21.34

Table 1: Top-1 validation errors on imbalanced IMDB review dataset on Bi-directional LSTM model with different approaches, where ErrorP means the validation errors of positive review, ErrorN means the validation errors of negative reviews, and ErrorM means mean errors.

Experiment	Original	Improved
ERM	29.41	**27.62**(-1.79)
b-SMOTE	NONE	68.98
RW+CB	27.88	**25.9** (-1.98)
RS+CB	29.43	28.45
LDAM	26.74	**24.23** (-2.51)
DRW	25.74	25.01
LDAM+DRW	23.12	**21.08** (-2.04)

Table 2: Validation errors of experiments reproduced with ResNet-32 on imbalanced CIFAR-10. The improved experiments with our proposed scheme (Section 6.3.3) reduces the error rate with all the experiments by different amount. Borderline-SMOTE is not in the original paper; we extended this experiment to compare different data re-balancing techniques (Section 6.3.4).

For IMDB, we noticed that the hyper parameters for experiments on IMDB was not given in the paper. However, we found that, by fine-tuning the bidirectional-LSTM model on its size of batch, number of epoch, and maximum number of features, we can achieve better results than the author's baseline as shown in the Table1.
For CIFAR-10, we first closely followed the author's work using the same hyper-parameters in the provided code. Our attempts on these experiments produced very similar to the reported results in the paper. The original results and our improved results are all reported in Table 2.

6.2 Baseline Track with IMDB

6.2.1 ERM/RS/RW/SMOTE baseline tuning experiments

Since the reference paper reported baseline mainly on the data level without explicitly giving the hyper-parameters, we constructed the step-by-step fine-tuning experiment based on four factors: number of training epoch, max number of feature, size of batch, and approach method. The default setting of the two-layer bidirectional LSTM with Adam optimizer is $BatchSize = 128$, $MaxFeature = 6000$, $Epoch = 5$, $Approach = ERM$. We conducted the experiment on size of batch with 32, 64, 128 as shown in Table 5. We observed that as the number of batch size increases, the overall mean error decreases. Then, we tried different max number of features ranging from 6000 to 12000 as shown in Table 3. We observed that a trade-off between the validation error for positive and negative set. As the maximum of features increases, the model tends to capture more features from the positive set and so can fit on the positive set better, and negative set worse. When the maximum of features reaches a certain threshold, the model tends to learn only the positive set and ignore the negative set, so that it produced a small loss on the positive set while a much bigger loss on the negative set. For these experiment results, we selected the final hyper-parameters set and visualized the training processing by plotting the loss and accuracy as shown in Figure 7.
Notice that we only included a detailed description of tuning experiments for the ERM method, however, the Re-sampling, Re-weighting, and SMOTE approachs followed the same procedure of fine-tuning. For Re-sampling and MOTE, we implemented the oversampling from imblearn[10]. For Re-weighting, we implemented sample weight from sklearn[11]. The top-1 validation and corre-

sponding configuration is reported in Table 1 and Appendix A.1. We noticed that the best model result in a least error rate of 16.48%, outperforming the all reported mean error in reference paper.

6.2.2 Investigation of linear model performance

In addition to the fine-tuning on the neural network, considering efficiency and previous work on IMDB set, we also implemented the linear models as shown in Table 4. The learning model is implemented from [11], and we implemented the grid search to find the optimal hyper-parameter. The performance of Logistic regression surprised us compared to other classifier, which all have mean error around 50%. The Logistic Regression with balanced re-weighting approach achieved 15.58% error rate, outperforming the referenced best score of 17.84%.

6.3 Baseline Track with CIFAR-10

6.3.1 ERM baseline tuning with step decay learning rate

In our ERM baseline experiment with long-tailed data, there is 29.41% errors on the validation set, which is similar to the error rate in the paper (29.64%). However, by plotting the loss change over the training duration Figure 3, we discovered that the training with the authors' hyper-parameters is extremely overfitted. After about 35 epochs, the validation loss started to increase, while the training loss continued decreasing until it reached almost zero after the entire training session. Therefore, we decided a new learning schedule in which we decrease the learning rate as soon as the validation loss does not show the tendency of decreasing. In this case, it will be around the 35th epoch. We adopted a step decay schedule: Train the initial 35 epochs with 0.1 learning rate; then decrease the learning rate to 0.01 and train the rest epochs. This approach is very effective as shown in Figure 3, which suggests an evident drop in both training and validation loss at the 35th epoch. The adjusted ERM error rate also drops significantly after the learning rate decay. The adjusted ERM was able to converge much faster and achieved 1.79% less validation errors than the original experiment.

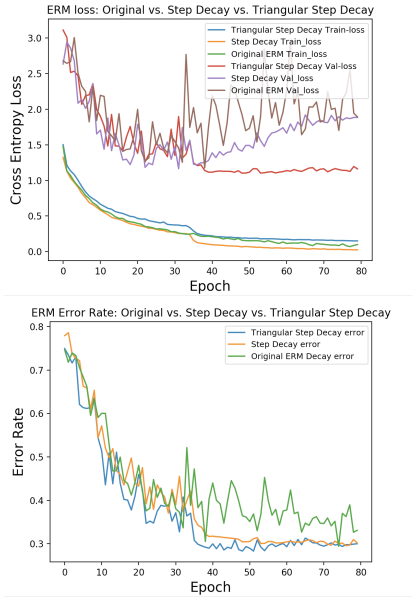

Figure 3: The left figure shows the training loss and validation loss comparison among the original ERM, step decay ERM and triangular step decay ERM. The right figure shows the error rate comparison among the original ERM, step decay ERM and triangular step decay ERM.

6.3.2 ERM baseline tuning with triangular learning rate policy

While the correctly planned step decay boosts the performance significantly, we further incorporate cyclical learning with the decayed learning rate. Cyclical learning introduces a dynamic learning rate within reasonable range, which is believed to be beneficial overall even though it might temporarily harm the network's performance[2]. We implemented the triangular learning policy to facilitate our step decayed learning rate. To estimate a good value for the step size, Leslie N. Smith [2] suggests that although the final accuracy results are quite robust to cycle length but experiments show that it is often good to set step size equal to 2 to 10 times the number of iterations in an epoch. In our case, we have long tailed ($\rho = 100$) imbalanced CIFAR-10 data with 20431 examples. The number of iterations per epoch is calculated by *total number of examples / batch size*, in our case, $20431/32 = 638$ iterations. We set our $stepsize = 8 * 638 = 5104$. Our step decay training schedule with triangular policy shows promising results (Figure3). Although the training loss with triangular policy is slightly higher, but the validation loss turned out to be a lot less, especially in the second stage of training. From the

validation error plot, we can also observe a slight performance advantage compared to pure step decay ERM.

6.3.3 Scheme for Baseline Improvement

Deep neural networks are expensive to train, which makes it very difficult to perform extensive experiments to find the optimal hyper-parameters. Based on Section 6.2 and Section 6.3, we propose a relatively forgiving and systematic scheme for running baselines, which consists of learning rate step decay and cyclical learning rate.

Estimate reasonable minimum and maximum boundary values: A simple way to estimate the learning rate bounding is to run the model for a few epochs with linearly increasing learning rate. The optimal learning rate is bounded between the learning rates that associate with the points when the accuracy starts to increase or decrease dramatically.

Learning rate scheduling: It is important to decay the learning rate at the right time with the right amount. If the learning rate is decayed to early, the model might converges to a bad local minimum. If the learning rate is decayed to late, the model is already overfitted and the training loss gets too small to continue training. Our approach it to run the model with a constant learning rate first and always record the losses of the last several iterations. Anneal the learning rate when the validation loss decreases too slowly or even starts increasing. This can be done multiple times until annealing the learning rate no longer improves the performance and stop training.

Enforce cyclical learning rate: Enforce a cyclic function bounding the learning range between the maximum and minimum boundaries provides substantial improvements in performance. This approach achieves improved accuracy without the need of experimental tuning and often in fewer iterations.

We deployed this scheme in all of the baseline experiments and observed improvements in some of them.

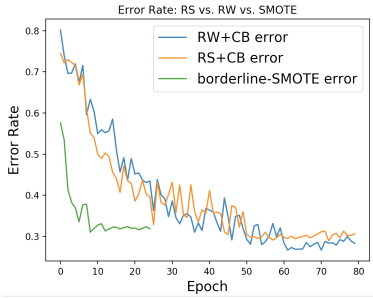

Figure 4: The validation error rate of long-tailed CIFAR-10 data trained with RW+CB, RS+CB, borderline-SMOTE

6.3.4 Re-weighting and Re-sampling

We experimented on several different data re-sampling and data-re-weighting approaches according to the author's experiments. We also extended the experiment with borderline-SMOTE over-sampling. Compare to the original SMOTE oversampling, borderline-SMOTE minimizes the intra-class variation and enlarges the inter-class variation. Moreover, the traditional re-weighting method is to re-weight by the inverse class frequency. Cui et al. [3] proposed a better scheme with re-weighting by the inverse effective number which is proved to yield better performance (RW+CB). Similarly, the inverse of effective number can also be used in data re-sampling (RS+CB). We compared our experiments with borderline-SMOTE, RW+CB and RS+CB, the error rate decay is shown in Figure4. Borderline-SMOTE converges a lot faster than the other two methods. With the long-tailed CIFAR-10 data, borderline-SMOTE drops the error rate rapidly, and the learning converges after only 10 epochs. RS+CB and RW+CB trains a lot slower compared to border-line SMOTE; however, with class balanced effective number, RS+CB and RW+CB have been seen a significant benefit after more iterations. To with RS, RW slightly out performs RS by around 2%.

6.3.5 Comments on how to find optimal stage transition points in DRW

LDAM+DRW approach performs great on imbalanced datasets compared to all the existing methods. Choosing the stage transition points carefully can boost the performance even more. In order to provide the second stage of DRW/DRS with

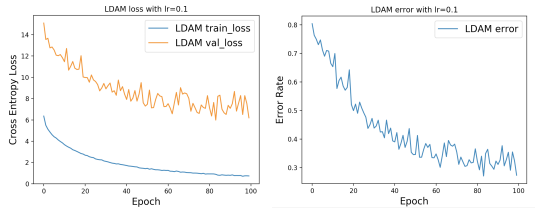

Figure 5: The performance behaviour of LDAM loss with constant learning rate 0.1

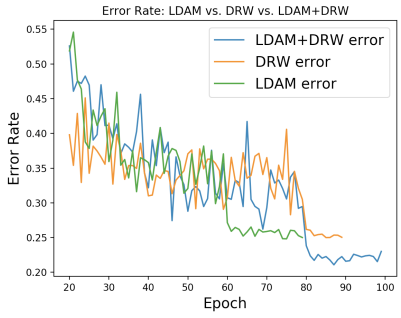

Figure 6: The error rate of LDAM+DRW campared with LDAM or DRW alone

as much potential as possible to continue learning, train loss cannot be too small before annealing the learning rate. Therefore, to find the optimal stage transition point, we start by training without DRW/DRS schedule. We applied LDAM loss on the model with constant learning rate 0.1 throughout the whole training session. The performance behaviour is plotted in Figure5. We observed that after 80 epochs, although the training loss is still decreasing, the loss decrements in training set no longer yields desirable decrements in either validation loss or error rate. Therefore, we decide the stage transition point to be at 80th epoch, where the learning rate is annealed by 100 times. Figure6 agrees with the authors claim that LDAM+DRW indeed outperforms LDAM or DRW alone.

7 Discussion and Conclusion

In this work, we performed extensive investigation on the baselines reported in the paper *Learning Imbalanced Datasets with Label-Distribution-Aware Margin Loss*. We particularly evaluated the baselines with a text classification task (IMDB) and an image classification task (CIFAR-10).
With manually created imbalanced IMDB data, firstly, we observed that traditional approach, logistic regression with a validation error of 15.85 %, can handle the imbalanced data elegantly. In logistic regression, for each feature contribute to the prevalence of target label, the model will assign an appropriate estimates to the particular feature. If the feature shows up rarely, then the resulting intercept will be very small; however, it still contribute to the prediction of label. Especially with the re-weighting, the sensitivity of the weak parameters estimate is further enhanced. Secondly, we observed that by fine-tuning, the bidirectional LSTM model achieved a validation error of 16.48%, better compared to their best score of 17.84%. Therefore, we concluded that, the LDAM-DRW model does not have always significant advantages over the classical approach on IMDB benchmark.
With the imbalanced CIFAR-10 data, instead of fine tuning the hyper-parameters, we proposed a general scheme that can be adopted in many situations. Our proposed scheme utilized learning rate decay and triangular policy, which often boosts the model performance without the need of extensive hyper-parameter tuning. This scheme is especially helpful for baseline improvement in deep neural networks. Since deep neural networks are expensive to train, researchers often spend more time on fine-tuning the proposed models than the baseline. Our approach makes it possible to achieve close to the optimal learning rate performance without extensive tuning. This schemes is applied to all CIFAR-10 baselines witch results in observable improvements among all of them. Additionally, our experiment results also agree with the authors' claim. We observed substantial advantage of deploying LDAM-DRW model on CIFAR-10.

8 Statement of Contribution

Doreen He: Worked on CIFAR-10 result reproduction and the write-up
Hehuimin Cheng: Worked on IMDB data set result reproduction and the write-up
Vitaly Kondulukov: Worked on baseline classic model result reproduction and the write-up

References

[1] Adrien Gaidon Nikos Arechiga Tengyu Ma Kaidi Cao, Colin Wei. Learning imbalanced datasets with label-distribution-aware margin loss.

[2] Leslie N. Smith. Cyclical learning rates for training neural networks.

[3] Tsung-Yi Lin Yang Song Serge Belongie Yin Cui, Menglin Jia. Class-balanced loss based on effective number of samples.

[4] M. Bing-Huan H. Han, W. Wen-Yuan. Borderline-smote: A new over-sampling method in imbalanced data sets learning.

[5] Vasile Palade Rukshan Batuwita. Efficient resampling methods for training support vector machines with imbalanced datasets.

[6] L. O. Hall W. P. Kegelmeyer N. V. Chawla, K. W. Bowyer. Smote: Synthetic minority oversampling technique.

[7] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning, 2018.

[8] Yoram Singer John Duchi, Elad Hazan. Adaptive subgradient methods for online learning and stochastic optimization.

[9] J. Chung Y. N. Dauphin, H. de Vries and Y. Bengio. Equilibrated adaptive learning rates for nonconvex optimization.

[10] Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, 18(17):1–5, 2017.

[11] Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, Robert Layton, Jake VanderPlas, Arnaud Joly, Brian Holt, and Gaël Varoquaux. API design for machine learning software: experiences from the scikit-learn project. In *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pages 108–122, 2013.

A Appendix

A.1 Parameter Setting and Results for IMDB experiments

ERM: BatchSize = 128, Max Feature = 8000, Epoch = 2. Re-samping: BatchSize = 128, Max Feature = 10000, Epoch = 4. Re-weighting: BatchSize = 128, Max Feature = 10000, Epoch = 5. SMOTE: BatchSize = 128, Max Feature = 10000, Epoch = 3. Adam Optimizer with default setting.

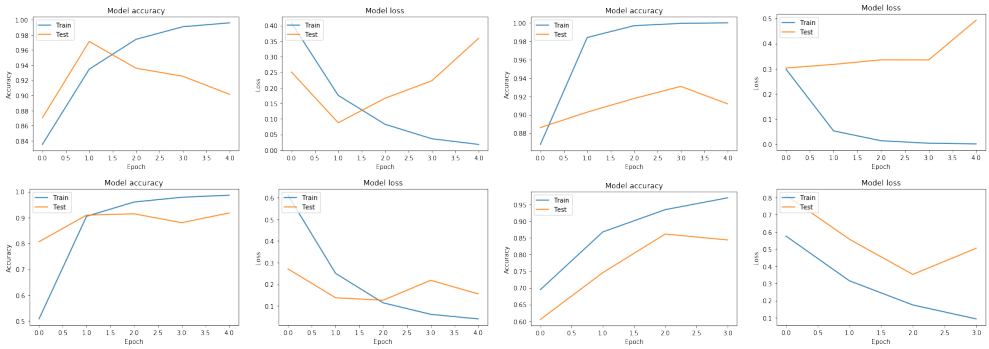

Figure 7: Accuracy and Loss plot for the best model under different approaches. The top left two plots correspond to ERM approach. The top right two plots correspond to Re-sampling approach. The bottom left two plots correspond to re-weighting. The bottom right two correspond to SMOTE approach.

MaxFeature	Error on positive reviews	Error on positive reviews	Mean Error
4000	6.07	43.96	25.02
6000	6.29	31.58	19.15
8000	**6.52**	**26.45**	**16.48**
10000	3.72	39.13	21.43
12000	0.14	90.49	45.32

Table 3: Top-1 validation errors on imbalanced IMDB review dataset on Bi-directional LSTM model for ERM approach with different maximum number of features based on the best model.

Classifier	Error on positive reviews	Error on positive reviews	Mean Error
Logistic Regression	**15.85**	**15.85**	**15.85**
XGBoost	0.00	100.00	50.00
KNN	0.00	100.00	50.00
NNaiveB	8.42	93.01	50.72

Table 4: Top-1 validation errors on imbalanced IMDB review dataset on Logistic Regression, XGBoost, Bernoulli Naive Bayes, and K-Nearst Neighboor.

BatchSize	Error on positive reviews	Error on positive reviews	Mean Error
32	3.31	41.47	22.39
64	8.50	30.24	19.37
128	**6.52**	**26.45**	**16.48**

Table 5: Top-1 validation errors on imbalanced IMDB review dataset on Bi-directional LSTM model for ERM approach with different batch set based on the best model.

A.2 Background

This section includes the algorithms described in [1].

Algorithm 1 Deferred Re-balancing Optimization with LDAM Loss

Require: Dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$. A parameterized model f_θ

Initialize the model parameters θ randomly
for $t = 1$ to T_0 **do**
  $\mathcal{B} \leftarrow \text{SampleMiniBatch}(\mathcal{D}, m)$ ▷ a mini-batch of m examples
  $\mathcal{L}(f_\theta) \leftarrow \frac{1}{m} \sum_{(x,y)\in\mathcal{B}} \mathcal{L}_{\text{LDAM}}((x, y); f_\theta)$
  $f_\theta \leftarrow f_\theta - \alpha \nabla_\theta \mathcal{L}(f_\theta)$ ▷ one SGD step
  Optional: $\alpha \leftarrow \alpha/\tau$ ▷ anneal learning rate by a factor τ if necessary

for $t = T_0$ to T **do**
  $\mathcal{B} \leftarrow \text{SampleMiniBatch}(\mathcal{D}, m)$ ▷ A mini-batch of m examples
  $\mathcal{L}(f_\theta) \leftarrow \frac{1}{m} \sum_{(x,y)\in\mathcal{B}} n_y^{-1} \cdot \mathcal{L}_{\text{LDAM}}((x, y); f_\theta)$ ▷ standard re-weighting by frequency
  $f_\theta \leftarrow f_\theta - \alpha \frac{1}{\sum_{(x,y)\in\mathcal{B}} n_y^{-1}} \nabla_\theta \mathcal{L}(f_\theta)$ ▷ one SGD step with re-normalized learning rate

Figure 8: The proposed LDAM-DRM algorithm from *Learning Imbalanced Datasets with Label-Distribution-Aware Margin Loss*[1]