LEARNING FROM IMBALANCED DATA DISTRIBUTION

By

Wentao Wang

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Computer Science—Doctor of Philosophy

2023

ABSTRACT

As a prominent component of artificial intelligence (AI), machine learning (ML) techniques play a significant role in the stunning achievement obtained by AI technologies in human society. ML techniques enable computers to leverage collected data to tackle various kinds of tasks in practice. However, more and more studies reveal that the capability of a ML model will be decreased dramatically if the distribution of collected data used for training this model is imbalanced. As imbalanced data distribution is widespread in many real-world applications, improving the performance of ML models under imbalanced data distribution has attracted considerable attention.

While a growing number of related works have been proposed to make ML models learn from imbalanced data more effectively, the study on this topic is far from complete. In this dissertation, I propose several studies to fill up the gaps in this direction. First, most existing data generation based works only consider the local distribution information within classes, while the global distribution is totally ignored. I demonstrate both global and local distribution information are important for producing high-quality synthetic data samples to balance the data distribution. Second, almost all existing studies assume that collected data samples are associated with noisy-free labels, and, hence, they cannot work well when annotated labels are noisy. I investigate the problem of learning from imbalanced crowdsourced labeled data and propose a novel framework as a solution with satisfactory performance. Third, currently the research on investigating the impact of imbalanced data distribution on the robustness of ML models is rather limited. To this end, I empirically verify the adversarial training (AT) approach alone cannot bring enough robustness for ML models under imbalanced scenarios while integrating the reweighting strategy with AT can be very helpful. In addition, I also propose an effective data augmentation based framework to benefit AT under imbalanced scenarios.

To my parents and entire family for their love and support.

ACKNOWLEDGEMENTS

This dissertation is impossible without invaluable help, support, and guidance I received from many great people during my Ph.D. study.

First and foremost, I would like to express my heartfelt thanks to my advisor Dr. Jiliang Tang. I am very grateful to him for providing me with a valuable opportunity that I can start a new journey in my life. During this unforgettable study journey, I have learned so much from him, ranging from discovering significant research problems, writing insightful papers, giving attractive presentations, collaborating with teammates, to establishing valuable career goals, etc. These priceless knowledge and experience I learned has benefited me a lot in my life and allowed me to achieve things I had never imagined. Dr. Tang is a role model for me to learn from and I feel honored to have been working with him.

I also would like to convey my deepest appreciation to other committee members of my dissertation, Dr. Hui Liu, Dr. Pan-Ning Tan and Dr. Yuying Xie, for their insightful comments and helpful suggestions.

During my Ph.D. study, I have had the pleasure and fortune of having so many intimate friends and colleagues. I would like to thank all of my colleagues from the Data Science and Engineering Lab for their selfless support and consistent encouragement: Zhikai Chen, Yingqian Cui, Dr. Jamell Dacon, Dr. Tyler Derr, Jiayuan Ding, Dr. Wenqi Fan, Kai Guo, Haoyu Han, Pengfei He, Dr. Jiangtao Huang, Dr. Wei Jin, Dr. Hamid Karimi, Hang Li, Juanhui Li, Yaxin Li, Dr. Haochen Liu, Hua Liu, Remy Liu, Dr. Xiaorui Liu, Dr. Yao Ma, Haitao Mao, Jie Ren, Harry Shomer, Wenzhuo Tang, Yuxuan Wan, Dr. Xiaoyang Wang, Dr. Xin Wang, Dr. Yiqi Wang, Dr. Zhiwei Wang, Hongzhi Wen, Han Xu, Kaiqi Yang and Dr. Xiangyu Zhao. I am also thankful for the collaboration from outside the Data Science and Engineering Lab: Wenbiao Ding, Yan Huang, Dr. Guoliang Li, Dr. Zitao Liu, Dr. Joseph Thekinen, Dr. Bhavani Thuraisingham, Dr. Suhang Wang and Guowei Xu.

Finally, I would like to thank my family for their love and support. I also dedicate this dissertation to Yiwei Ma for supporting me all the way.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Motivation

As a prominent component of artificial intelligence (AI), machine learning techniques play an important role in the great successes achieved by AI technologies in human world. The core objective of machine learning is to instruct computers to leverage collected data to tackle diverse tasks [75]. Machine learning techniques have been applied into a wide range of applications, which facilitate people's daily lives greatly while also improving productivity in various sectors effectively. For instances, in e-commerce, recommender systems [6] can provide personalized product recommendations to customers, which helps customers find their interested products more efficiently and hence brings more profits to e-commerce platforms; in information security, face recognition systems [53] are able to confirm people's identities by matching their faces against faces stored in database, which accelerates the verification process when people accessing confidential data and also supplies an enhanced security for protecting these confidential data at the same time; in healthcare, medical image analysis methods [65] make diagnosis by analyzing medical images, which leads to reduced costs for patients and improved efficiency for healthcare organizations.

Despite the huge potential of developing advanced machine learning algorithms to handle more complex tasks and enhance the power of AI in human world, recent studies [8] reveal that the capability of a machine learning model will be decreased dramatically if the quality of data used for training this model is low. For supervised classification algorithms who utilize data samples with corresponding class labels to train a model to predict class information of target data samples, imbalanced data distribution can be regarded as one common type of low-quality data. The imbalanced data distribution refers to the case that some classes have exceedingly large number of data samples while others have very small amount of data samples in the data set. Since most supervised classification algorithms are developed based on a common assumption that each class in the training data set has a relatively equal number of data samples, their trained models tend to be overwhelmed by classes with large training samples while ignoring classes with small training

samples [16] in the training phase and hence cannot obtain satisfactory classification performance on these ignored classes in the inference phase.

Considering imbalanced data distribution is widespread in many applications and aforementioned negative impacts brought by it for machine learning algorithms, in the past few decades, many efforts have been devoted to making models learn from imbalanced data more effectively. Although many related works have been proposed, the study on this topic is far from complete. First, when generating synthetic data samples, most existing data generation based works only consider the local distribution information within classes with amount of data samples, while the global distribution is totally ignored. Hence, the quality of synthetic data samples generated by these work cannot be guaranteed. Second, almost all existing studies assume that collected data samples are associated with noisy-free labels. Therefore, they cannot deal with a more complicated but realistic scenario that the annotated labels are noisy. For example, multiple crowd works may be invited to annotated labels for collected data samples and hence the annotated labels can be very inconsistent and noisy. Third, the majority of all previous studies only focus on improving the prediction accuracy of models under imbalanced data distribution, while the research on investigating the impact of imbalanced data distribution for the robustness of trained models is rather limited. Recently, the robustness of machine learning models has attracted increasing attention when deploying machine learning models into real-world applications.

In this dissertation, I present several studies to fill up the gaps in terms of three aforementioned perspectives. First, I focus on mitigating imbalanced data distribution through generating synthetic data samples. I will demonstrate both global and local distribution information are important for producing high-quality synthetic data samples. Second, I investigate the problem of learning from imbalanced crowdsourced labeled data, which is a more realistic and challenging scenario in the real world. I will propose a novel framework for training a discriminative model on imbalanced crowdsourced labeled data directly with satisfactory prediction performance. Third, I study the model robustness problem under imbalanced data distribution. I will empirically verify that the adversarial training approach alone cannot bring enough robustness for models under imbalanced

scenarios while integrating reweighting strategy with adversarial training can be very helpful. In addition, I will also present an effective data augmentation based framework to befit adversarial training under imbalanced scenarios

1.2 Contributions

The major contributions of this dissertation are summarized as follows:

- I conduct research on three important but less-explored topics about learning from imbalanced data distribution: (1) generating high-quality synthetic data, (2) learning from imbalanced crowdsourced labeled data, and (3) improving model robustness;

- In chapter 2, I first identify the importance of both global and local distribution information in data generation approaches for imbalanced data and then propose a novel framework to generate more realistic synthetic data samples under imbalanced data distribution by utilizing both global and local distribution information;

- In chapter 3, I present a novel framework to obtain a discriminative model that can achieve good prediction performance on all classes involved in the data set by training on imbalanced crowdsourced labeled data directly;

- In chapter 4, I first empirically discover two major differences between naturally trained models and models trained by adversarial training approach under imbalanced data distribution and then propose a new framework to improve model robustness under imbalanced data distribution.

- In chapter 5, I present an effective framework to augment imbalanced training data so that the robustness of models can be boosted by applying adversarial training into the training phase.

CHAPTER 2

GLOBAL-AND-LOCAL AWARE DATA GENERATION FROM IMBALANCED DATA DISTRIBUTION

In many real-world classification applications such as fake news detection, the training data can be extremely imbalanced, which brings challenges to existing classifiers as the majority classes dominate the loss functions of classifiers. Oversampling techniques such as SMOTE are effective approaches to tackle the class imbalance problem by producing more synthetic minority samples. Despite their success, the majority of existing oversampling methods only consider local data distributions when generating minority samples, which can result in noisy minority samples that do not fit global data distributions or interleave with majority classes. Hence, in this chapter, we study the class imbalance problem by simultaneously exploring local and global data information since: (i) the local data distribution could give detailed information for generating minority samples; and (ii) the global data distribution could provide guidance to avoid generating outliers or samples that interleave with majority classes. Specifically, we propose a novel framework GL-GAN, which leverages the SMOTE method to explore local distribution in a learned latent space and employs GAN to capture the global information, so that synthetic minority samples can be generated under even extremely imbalanced scenarios. Experimental results on diverse real data sets demonstrate the effectiveness of our GL-GAN framework in producing realistic and discriminative minority samples for improving the classification performance of various classifiers on imbalanced training data.

2.1 Chapter Introduction

The classification performance heavily relies on the quality and quantity of the training data [49]. However, in many real-world applications, due to some practical concerns such as privacy and time cost, only limited labeled data can be collected. Meanwhile, such data could be imbalanced. Specifically, some classes have significantly larger number of data samples while others have very limited amount of data, which is called class imbalance problem [44]. For instance, in fake news detection [88], the majority of news in the collected data are true news while only a small portion

of news are fake news. The imbalanced data has negative impacts on the classifier training since the standard classifiers tend to be overwhelmed by the majority classes while ignoring the minority classes [16]. Furthermore, even though minority classes may only take extremely small ratio of one data set, for some applications like medical diagnosis, misclassifying a minority class sample is usually more severe than misclassifying a majority one [68].

Oversampling has been proven to be an effective way to alleviate the class imbalance problem by oversampling minority samples into the imbalanced data set [70]. As one of the most popular oversampling methods, Synthetic Minority Over-sampling Technique (SMOTE) [14] generates new synthetic minority samples by performing linear interpolation operations between existing minority samples and their nearest neighbors within the same class. As shown in Figure 2.1, by applying the SMOTE method, new synthetic minority samples are generated along with the linear interpolation between two existing minority samples.

Despite the success of SMOTE and its variants [35, 39], they still face some challenges. First, SMOTE-based methods only consider the local neighbor relationship of each minority sample, while the global distribution is totally ignored. Without considering the global distribution of the given data, the generated minority samples could not fit the real data distribution. For instance, the generated samples in Figure 2.1 are either located on the null space of the given data samples or interleaved with majority data samples. Second, the interpolation operations performed by these methods on raw feature space may not generate realistic data samples. For instance, for the given text data which lies in discrete space, SMOTE-based methods cannot guarantee their generated texts are readable.

Therefore, in this chapter, we study the class imbalance problem by simultaneously exploring both global and local information. The local data distribution provides detailed local information for generating minority samples; and the global data distribution provides guidance from a global view to avoid generating samples that interleave with majority samples or fall in the null space of the given data. We are faced with two challenges: (i) how to explore global data distribution for minority sample generation; and (ii) how to simultaneously leverage global and local distribution information

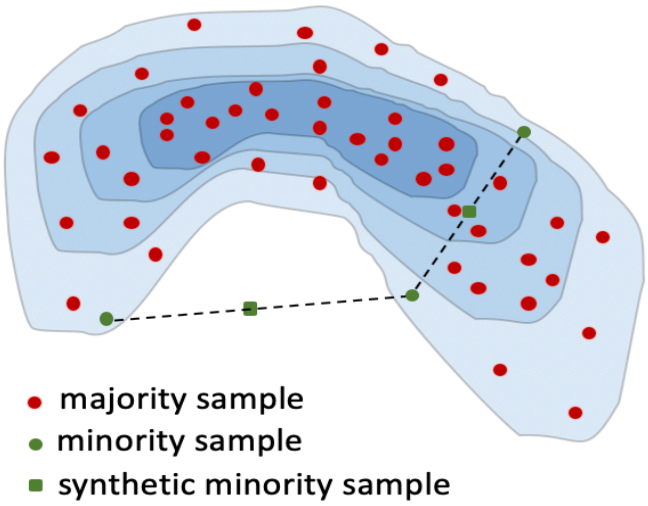


Figure 2.1 An example of imbalanced data and SMOTE method. The synthetic minority samples are generated along the dash line between two minority samples.

to generate realistic and discriminative synthetic minority samples. Recently, generative adversarial learning [31] has shown promising results in generating realistic data samples [31, 74] through estimating the latent global data distribution, which paves us a way to solve these two challenges. Hence, we propose a novel framework which leverages oversampling techniques to capture local data structure and generative adversarial learning to explore global data distribution. The contributions of this chapter are summarized below:

- We identify the importance of both global and local distribution information in tackling the class imbalance problem.
- We propose a novel generative adversarial framework, GL-GAN, to generate realistic and discriminative minority samples by exploring both global and local distributions.
- We conduct extensive experiments on diverse real data sets to demonstrate the effectiveness of GL-GAN on alleviating the class imbalance problem.

2.2 Related Work

In many real world applications, we are faced with class imbalance problem. The popularity of class imbalance has attracted increasing attention, and, various kinds of effective approaches have

been proposed in the last few decades. Existing works for tackling the class imbalance problem can be roughly classified into three categories[55]: (i) data-level methods, which modify the class distribution by adding or removing samples from training set; (ii) algorithm-level methods, which modify the existing algorithm to adapt imbalanced scenarios; (iii) and hybrid methods, which combine the advantages of two previous categories. Our GL-GAN is a data-level method.

Undersampling [110, 68] and oversampling [14, 35, 39] are two fundamental data-level solutions. Briefly, undersampling approaches downsize the majority class by removing majority samples, while oversampling approaches upsize the minority class by generating minority samples [66]. Oversampling with replacement, also called random oversampling [30], is the simplest oversampling approach that randomly duplicates existing minority samples to augment the minority class. However, the random oversampling method often makes the decision boundary of the classifier smaller and causes the classifier to over-fit [35]. As an improved approach, SMOTE [14] inflates the minority class by producing synthetic minority samples instead of duplicating existing minority samples. Based on the SMOTE method, several variants, such as borderline-SMOTE1 and borderline-SMOTE2 [35] and ADASYN [39], have been proposed to achieve better performances in the past few years. Different from SMOTE-based methods that utilize Euclidean distance to perform interpolation operations, some recent work [2, 86] introduced Mahalanobis distance into synthetic minority samples generation process and achieved good performance on classifier training.

Recently, more and more researchers have been attracted by the generative adversarial learning due to its great power on generating different kinds of realistic synthetic data samples. The pioneer work introduced by [31] presented Generative Adversarial Networks (GAN) to learn the real data distribution through a minimax game between a generator G and a discriminator D. The generator G produces synthetic samples to fool the discriminator D, while the discriminator D judges whether the input samples come from the generator or from the real data set. These two components fight against each other and improve themselves gradually [29]. In the perfect equilibrium, the generator G is able to capture the global distribution information of real training data and generate synthetic samples following this distribution [20]. Due to the great ability of generative adversarial learning

models on generating realistic data samples, some recent research applied generative adversarial learning into the class imbalance problem. For instance, conditional GAN [74] is adopted in [24] for producing minority samples effectively. BAGAN [72], is a data augmentation model that can alleviate the class imbalance problem by modifying the discriminator D in the traditional GAN. However, all aforementioned models used in solving the class imbalance problem take the random noise as the input of generator G, which may bring a lot of uncertainty during the model training phase. Moreover, the local structure of training minority samples is not explored by these models, so some generated synthetic samples may close to the decision boundary and thus hard to be utilized to train a classifier.

Our GL-GAN is inherently different from existing works. We simultaneously explore the global and local information by leveraging local-based oversampling techniques and generative adversarial learning models. Therefore, GL-GAN can overcome the drawbacks of the oversampling techniques and utilize the power of the generative adversarial models to produce more realistic and discriminative synthetic minority data samples.

2.3 The Proposed Framework

In this chapter, we focus on the binary class imbalance problem. Given an imbalanced sample set $\mathcal{X}_{org}$ containing a majority sample set $\mathcal{X}_{maj}$ and a minority sample set $\mathcal{X}_{min}$, our goal is to generate a set of realistic and discriminative synthetic minority samples $\mathcal{X}_{syn}$ so that, comparing with training only on the original imbalanced sample set $\mathcal{X}_{org}$, the classification performance of classifiers can be greatly improved by training on the balanced augmentation sample set $\mathcal{X}_{org} \cup \mathcal{X}_{syn}$.

As shown in Figure 2.2, our GL-GAN is composed of two modules, local structure exploration and global distribution learning. The former is designed for generating latent representations of minority samples through exploring the local distribution information, and the latter aims to produce realistic and discriminative minority samples that can fit the global distribution. Next, we will introduce details of each module.

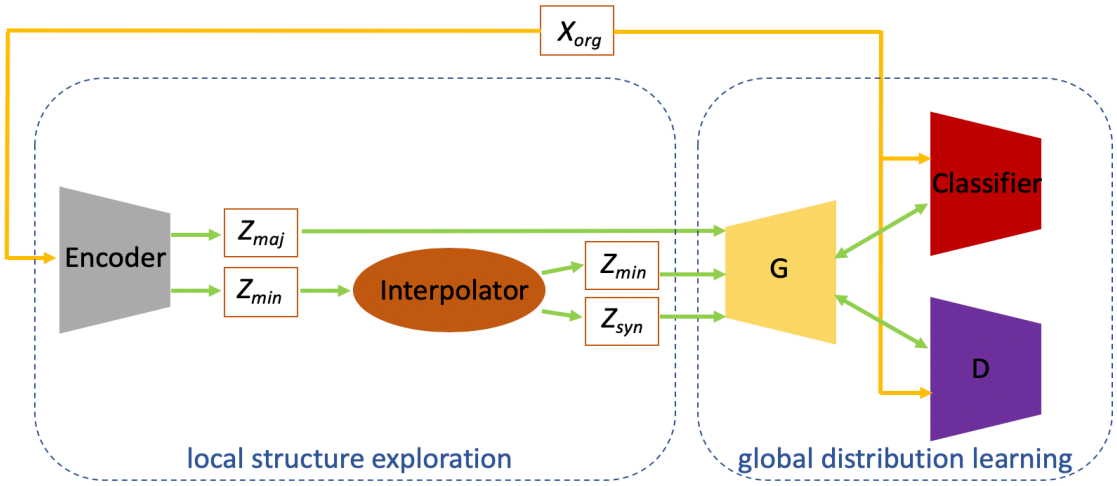


Figure 2.2 An overview of GL-GAN.

2.3.1 Local Structure Exploration

The local structure exploration module consists of two components, i.e., an encoder E and a local data representation interpolator I, specifying for two different tasks separately.

2.3.1.1 Discriminative Representation Learning

In many cases, directly generating synthetic data samples in raw feature space by local-based oversampling techniques such as SMOTE may cause several problems.

Firstly, as we demonstrated before, these methods cannot generate realistic synthetic samples for some specific data types like text data. Secondly, the generated minority data samples may interleave with majority samples. This motivates us to first learn discriminative latent representations of the raw data, then exploit the local data structure in the learned latent space. The advantages of doing this are as follows: (i) By learning a low-dimensional latent representation, we can preserve the most important information of the data while drop some noisy information; and (ii) During the latent representation learning process, we can enforce the latent representations of data samples belonging to the same class to be closed to each other.

Deep autoencoders have been proved to be an effective way to extract important information from high-dimensional data using low-dimensional representations [84]. Typically, an autoencoder consists of two components: an encoder E and a decoder Q. The encoder E takes the high-dimensional data as input and maps them to the corresponding latent representations. The decoder

Q recovers these learned latent representations back to the raw feature space. The goal of training an autoencoder is to minimize the reconstruction error between the input data and the reconstructed data produced by the decoder Q, which can be defined as

$$\mathcal{L}_{rec}(Q\left(E\left(\mathcal{X}_{org}\right)\right), \mathcal{X}_{org}) = \frac{1}{|\mathcal{X}_{org}|} \sum_{x_i \in \mathcal{X}_{org}} \|Q\left(E\left(x_i\right)\right) - x_i\|_2^2. \tag{2.1}$$

In our GL-GAN framework, we propose to embed the given real data samples into a latent space with majority samples in one cluster and minority samples in another cluster, and these two clusters should be far-away from each other. To do that, we aim to reduce the interleaving between the synthetic generated minority samples and the given majority samples. Formally, this process can be described by

$$\begin{aligned} \mathcal{L}_{clu} = & \frac{1}{|\mathcal{X}_{maj}|} \sum_{x_i \in \mathcal{X}_{maj}} \|E(x_i) - \overline{z}_{maj}\|_2^2 + \frac{\lambda_1}{|\mathcal{X}_{min}|} \\ & \sum_{x_i \in \mathcal{X}_{min}} \|E(x_i) - \overline{z}_{min}\|_2^2 - \lambda_2 \|\overline{z}_{maj} - \overline{z}_{min}\|_2^2, \end{aligned} \tag{2.2}$$

where $\overline{z}_{maj}$ and $\overline{z}_{min}$ are mean of the latent representations of majority sample set X_{maj} and minority sample set X_{min}, respectively. λ_1 and λ_2 are two hyper-parameters controlling the weights. Starting from here, we use Λ or λ to represent hyper-parameters.

Therefore, the autoencoder in our GL-GAN can be trained by minimizing the following loss function:

$$\mathcal{L}_A = \mathcal{L}_{rec} + \Lambda_1 \mathcal{L}_{clu} + \Lambda_2 R(\theta). \tag{2.3}$$

Here $R(\theta)$ is the regularizer of the model parameters θ. Once the autoencoder is trained well, the latent representation of sample x_i can be given as $z_i = E(x_i)$.

2.3.1.2 Local-based Data Generation

With the learned latent representations, we can generate synthetic minority samples in the latent space by exploring the local structure of the sample set $\mathcal{Z}_{min}$, which is the latent embedding of the minority sample set $\mathcal{X}_{min}$. In our GL-GAN, we adopt SMOTE as the implementation of the local data interpolator I because of its simplicity.

For any minority sample $z_i \in \mathcal{Z}_{min}$, SMOTE 1) discovers k nearest neighbors $\{z_i^1, z_i^2, \ldots, z_i^k\}$ of z_i within the same minority class set $\mathcal{Z}_{min}$, 2) randomly picks up any one nearest neighbor z_i^n ($n \in [1, k]$) from the set $\{z_i^1, z_i^2, \ldots, z_i^k\}$ and chooses a random number $\eta \in [0, 1]$. Hence, a new synthetic minority sample z_i' could be created by $z_i' = z_i + \eta \left(z_i^n - z_i\right)$. The second step can be repeated N times, and, finally, $N \times |\mathcal{Z}_{min}|$ synthetic minority samples will be generated when executing the same process on every minority sample in $\mathcal{Z}_{min}$. After the synthetic minority sample set Z_{syn} is obtained, we can get a balanced augmentation sample set $\mathcal{Z} = \mathcal{Z}_{maj} \cup \mathcal{Z}_{min} \cup \mathcal{Z}_{syn}$ in the latent space.

2.3.2 Global Distribution Learning

For making the generated minority samples in $\mathcal{Z}_{syn}$ more realistic and discriminative, we introduce a generative adversarial learning model to learn the global information of given samples and modify samples in $\mathcal{Z}_{syn}$ accordingly.

2.3.2.1 Discriminator D

The role of the discriminator D is to differentiate if a data sample is real or fake. For a data sample who comes from the given real data set, the discriminator D labels it as a real sample. If a data sample is synthetically generated by the generator G, it will be classified as a fake sample. The discriminator D and the generator G fight against each other and improve themselves gradually. The loss function for training the discriminator D can be written as

$$\mathcal{L}_D = \frac{1}{|\mathcal{X}_{org}|} \sum_{x_i \in \mathcal{X}_{org}} \|D(x_i) - 1\|_2^2 + \frac{\lambda}{|\mathcal{Z}|} \sum_{z_i \in \mathcal{Z}} \|D(G(z_i)) - 0\|_2^2. \tag{2.4}$$

In equilibrium, the discriminator D cannot find the difference between real and synthetic samples, which means the quality of synthetic data generated by the generator G are approximate to the real data.

2.3.2.2 Classifier C

For making sure that generated data samples can have expected labels, we introduce a classifier C in our GL-GAN. Specifically, the classifier C also takes both real samples and synthetic samples generated by the generator G as input. Since the input of G in GL-GAN is the balanced augmentation

sample set $\mathcal{Z}$, every output of the generator G, i.e. $G(z_i)$, has its corresponding label. The classifier C works on labeled data samples and makes classification for them. The loss function for training the classifier C in our GL-GAN is

$$\mathcal{L}_C = \frac{1}{|\mathcal{X}_{org}|} \sum_{x_i \in \mathcal{X}_{org}} \|C(x_i) - \Gamma_{x_i}\|_2^2 + \frac{\lambda}{|\mathcal{Z}|} \sum_{z_i \in \mathcal{Z}} \|C(G(z_i)) - \Gamma_{z_i}\|_2^2. \tag{2.5}$$

Here Γ_{x_i} and Γ_{z_i} are true labels of real sample x_i and latent representation z_i, respectively. By introducing the classifier C into the traditional GAN, the generator G is forced to produce synthetic samples which can be classified by C correctly.

2.3.2.3 Generator G

Different from the traditional generator G that takes a set of random noise following some prior distribution as input, during the model training phase, the generator G in our GL-GAN is fed with the balanced augmentation sample set $\mathcal{Z}$. Since there are two types of latent representations in $\mathcal{Z}$, i.e., the latent representations of real samples in $\mathcal{Z}_{maj}$ and $\mathcal{Z}_{min}$, denoted as $\mathcal{Z}_{org} = \mathcal{Z}_{maj} \cup \mathcal{Z}_{min}$, and the latent representations of synthetic samples in $\mathcal{Z}_{syn}$, the generator G should be able to project latent representations $\mathcal{Z}_{org}$ back to the raw feature space as well as produce synthetic data samples that can fool the discriminator D. Therefore, the loss for training generator G includes three different types: the reconstruction loss $\mathcal{L}_{rec}$ for mapping latent representations $\mathcal{Z}_{org}$ back to the raw feature space, the discriminator loss $\mathcal{L}_{(G,D)}$ produced by the discriminator D for evaluating the difference between the real data samples and data samples generated by G, and the classifier loss $\mathcal{L}_{(G,C)}$ brought by the classifier C for making classification on the generated data samples of G. Formally, the loss function for training the generator G in our GL-GAN can be defined as

$$\begin{aligned}
\mathcal{L}_G &= \mathcal{L}_{rec}(G(\mathcal{Z}_{org}), \mathcal{X}_{org}) + \lambda_1 \mathcal{L}_{(G,D)} + \lambda_2 \mathcal{L}_{(G,C)} \\
&= \frac{1}{|\mathcal{X}_{org}|} \sum_{x_i \in \mathcal{X}_{org}, z_i \in \mathcal{Z}_{org}} \|G(z_i) - x_i\|_2^2 + \frac{\lambda_1}{|\mathcal{Z}|} \\
&\sum_{z_i \in \mathcal{Z}} \|D(G(z_i)) - 1\|_2^2 + \frac{\lambda_2}{|\mathcal{Z}|} \sum_{z_i \in \mathcal{Z}} \|C(G(z_i)) - \Gamma_{z_i}\|_2^2.
\end{aligned} \tag{2.6}$$

After the whole framework is trained well, the generator G is able to produce a set of realistic and discriminative synthetic minority samples.

Algorithm 2.1 The algorithm of GL-GAN.

Input: an imbalanced sample set X_{org}

Initialize the parameters of autoencoder.
Pre-train the autoencoder to obtain the latent representations $\mathcal{Z}_{org} = \mathcal{Z}_{maj} \cup \mathcal{Z}_{min}$ of $\mathcal{X}_{org}$.
Apply SMOTE method for $\mathcal{Z}_{min}$ to get the synthetic minority sample set $\mathcal{Z}_{syn}$.
Form a balanced augmentation sample set $\mathcal{Z} = \mathcal{Z}_{maj} \cup \mathcal{Z}_{min} \cup \mathcal{Z}_{syn}$ in the latent space.
repeat
  for discriminator-epochs **do**
    Train the discriminator D with augmented latent sample set $\mathcal{Z}$ and real sample set $\mathcal{X}_{org}$ (Sec. 2.3.2.1).
  end for
  for classifier-epochs **do**
    Train the classifier C with augmented latent sample set $\mathcal{Z}$ and real sample set $\mathcal{X}_{org}$ (Sec. 2.3.2.2).
  end for
  for generator-epochs **do**
    Train the generator G (Sec. 2.3.2.3).
  end for
until model convergence

2.3.3 Objective Function of GL-GAN

With local structure exploration module and global distribution learning module introduced above, the final objective function of GL-GAN is given as:

$$\min_{\theta_G,\theta_C} \max_{\theta_D} \mathcal{L}_{rec}(G(\mathcal{Z}_{org}), \mathcal{X}_{org}) + \Lambda_1 \mathcal{L}_{(G,D)} + \Lambda_2 \mathcal{L}_{(G,C)} \tag{2.7}$$

where θ_G, θ_C and θ_D are the parameters of generator G, classifier C and discriminator D, respectively.

2.3.4 Algorithm

In this subsection, we present our GL-GAN framework in Algorithm 2.1.

As shown in Algorithm 2.1, we train the autoencoder part at first to make sure the autoencoder could map the input data samples into two far-way clusters in the latent space. After pre-training the autoencoder, we utilize the encoder E to obtain the latent representations of the input data samples. Then, the local data interpolator I can be applied in the learned latent space to generate a set of synthetic minority samples within the same cluster. In order to train the generative adversarial learning part more effectively, we use the knowledge learned by the pre-trained autoencoder to

initialize the generative model. Specifically, the discriminator D and the classifier C have the same architecture with the encoder E except both D and C have one more layer. The last layer of the discriminator D is a dense layer with a softmax activation function for producing binary outputs and the last layer of the classifier C is a dense layer for producing classification results. The parameters learned by the encoder E will be used to initialize the discriminator D and the classifier C during the generative model training phase. Similarly, the generator G is initialized by the weight parameters learned by the decoder Q since they have the same architecture.

2.4 Experiment

In this section, we conduct experiments to verify the effectiveness of our proposed GL-GAN framework. We aim at answering the following two questions:

- Can the proposed GL-GAN framework generate discriminative minority samples for improving the classification performance of imbalanced data?
- What is the impact of each module of GL-GAN?

We begin by introducing the data sets and experimental settings, then we compare GL-GAN with several state-of-the-art related methods on the classification task to answer the first question. We then analyze the impact of each module of GL-GAN to answer the second question.

2.4.1 Experimental Settings

In order to test how the generated synthetic samples alleviate the binary class imbalance problem, we utilize the classification performance of different classifiers training on various augmented sample sets as the evaluation indicator.

2.4.1.1 Data Sets

The experiments are conducted on five real data sets, i.e., USPS, Sensorless Drive Diagnosis, Gas Sensor Array Drift, Madelon and Gisette. Sensorless Drive Diagnosis and Gas Sensor Array Drift are publicly from the UCI data repository[1] and the rest three can be obtained from Feature Selection data repository[2]. Since all these five data sets are class balanced, we construct the

[1] https://archive.ics.uci.edu/ml/index.php
[2] http://featureselection.asu.edu/datasets.php

Table 2.1 Statistical information of imbalanced data sets.

Data Set	# Features	# Majority	# Minority
USPS	256	744	7
Sensorless Drive Diagnosis	48	4256	42
Gas Sensor Array Drift	128	1549	15
Madelon	500	1040	10
Gisette	5000	2800	28

imbalanced data set for each of them according to the following three steps. Firstly, we randomly choose one class as majority class and another one as minority class to obtain a balanced binary data set. Then we divide 80% data samples of the balanced binary class data set as the candidates set and the rest as the test set. Lastly, we artificially imbalance the candidates set to form the imbalanced data set by utilizing a predefined imbalanced ratio r. For instance, if $r = 0.01$, then 99% minority samples will be removed from the candidates set so that the ratio of the minority samples to the majority samples in the imbalanced data set is 0.01. Table 2.1 provides the statistical information of five imbalanced data sets obtained by the aforementioned three steps when the imbalanced ratio $r = 0.01$.

2.4.1.2 Classifiers

Since our goal is to generate synthetic minority samples for improving the classification performance, we introduce several classifiers to help to evaluate the quality of the generated samples. Three representative classifiers, i.e., Multi-layer Perceptron (MLPClassifier), Linear Support Vector Classification (LinearSVC) and AdaBoost are adopted in our experiments. We train these classifiers on the training sets augmented by the synthetic minority samples generated by our model or baselines and test them on the corresponding test data sets. All these three classifiers are implemented by the *scikit-learn* package[3] in Python, and we use their default settings in all experiments.

2.4.1.3 Evaluation Metrics

For measuring the classification performance of classifiers, we introduce three different metrics, macro F1-score, micro F1-score and Matthews correlation coefficient (MCC) [73] into our experiments. The value of MCC is in the range $[-1, 1]$, in which MCC = 1 indicates a perfect prediction,

[3] https://scikit-learn.org/stable/index.html

Table 2.2 Classification performance of classifiers on the USPS data set.

Evaluation		Method							
Classifier	Metrics	Imbalanced	Random	SMOTE	MDO	NRAS	SWIM	BAGAN	GL-GAN
MLPClassifier	macro F1	0.7712	0.8840	0.8825	0.8914	0.8415	0.6443	0.8208	**0.8937**
	micro F1	0.7969	0.8899	0.8887	0.8947	0.8503	0.6595	0.8356	**0.8985**
	MCC	0.6232	0.7839	0.7823	0.7858	0.7022	0.4731	0.6872	**0.7990**
LinearSVC	macro F1	0.8473	0.8580	0.8580	0.8834	0.8408	0.6184	0.8836	**0.8912**
	micro F1	0.8589	0.8681	0.8680	0.8865	0.8497	0.6380	0.8896	**0.8957**
	MCC	0.7346	0.7510	0.7510	0.7683	0.7010	0.4440	0.7838	**0.7913**
AdaBoost	macro F1	0.8024	0.7878	0.8036	0.8436	0.7883	0.8900	0.7848	**0.8920**
	micro F1	0.8218	0.8095	0.8221	0.8558	0.8098	0.8906	0.8077	**0.8966**
	MCC	0.6689	0.6441	0.6662	0.7291	0.6444	0.7865	0.6433	**0.7942**

Table 2.3 Classification performance of classifiers on the Sensorless Drive Diagnosis data set.

Evaluation		Method							
Classifier	Metrics	Imbalanced	Random	SMOTE	MDO	NRAS	SWIM	BAGAN	GL-GAN
MLPClassifier	macro F1	0.7697	0.8642	0.8700	0.8580	0.8510	0.8439	0.9078	**0.9334**
	micro F1	0.7809	0.8666	0.8722	0.8608	0.8542	0.8476	0.9086	**0.9338**
	MCC	0.6251	0.7608	0.7699	0.7513	0.7406	0.7299	0.8310	**0.8755**
LinearSVC	macro F1	0.8195	0.8425	0.8425	0.8455	0.8420	0.8435	**0.9281**	0.8714
	micro F1	0.8250	0.8462	0.8462	0.8490	0.8457	0.8471	**0.9285**	0.8735
	MCC	0.6939	0.7277	0.7277	0.7322	0.7269	0.7292	**0.8659**	0.7721
AdaBoost	macro F1	0.8962	0.9683	0.9686	0.9955	0.9835	0.9924	0.9817	**0.9959**
	micro F1	0.8973	0.9683	0.9687	0.9955	0.9835	0.9924	0.9817	**0.9959**
	MCC	0.8119	0.9385	0.9392	0.9910	0.9676	0.9849	0.9638	**0.9918**

Table 2.4 Classification performance of classifiers on the Gas Sensor Array Drift data set.

Evaluation		Method							
Classifier	Metrics	Imbalanced	Random	SMOTE	MDO	NRAS	SWIM	BAGAN	GL-GAN
MLPClassifier	macro F1	0.3879	0.7880	0.8116	0.6540	0.7182	0.6974	**0.8891**	0.8881
	micro F1	0.5376	0.8003	0.8201	0.6833	0.7424	0.6985	**0.8908**	0.8884
	MCC	0.1077	0.6518	0.6832	0.4819	0.5592	0.3967	**0.7933**	0.7782
LinearSVC	macro F1	0.8580	0.9270	0.9296	0.6588	0.9216	0.3822	0.9290	**0.9694**
	micro F1	0.8619	0.9270	0.9296	0.6866	0.9216	0.5126	0.9296	**0.9695**
	MCC	0.7511	0.8560	0.8610	0.4871	0.8445	0.1606	0.8666	**0.9391**
AdaBoost	macro F1	0.3825	0.4620	0.4739	0.5299	0.5295	0.4795	0.4995	**0.5976**
	micro F1	0.5255	0.5418	0.5352	0.5963	0.6082	0.5445	0.5352	**0.6082**
	MCC	0.0689	0.0940	0.0690	0.3423	0.3174	0.0954	0.0653	**0.2181**

MCC = 0 means the prediction made by a classifier is no better than the random prediction and MCC = -1 represents total wrong between the prediction and the observation.

2.4.2 Effectiveness Evaluation

For evaluating the effectiveness of our GL-GAN framework on alleviating the binary class imbalance problem, we compare the quality of the synthetic samples generated by GL-GAN with several representative and state-of-the-art oversampling methods, including: 1) Imbalanced, which

Table 2.5 Classification performance of classifiers on the Madelon data set.

Evaluation		Method							
classifier	Metrics	Imbalanced	Random	SMOTE	MDO	NRAS	SWIM	BAGAN	GL-GAN
MLPClassifier	macro F1	0.3333	0.3346	0.3364	0.4513	0.4440	0.4088	0.3376	**0.4821**
	micro F1	0.5000	0.5006	0.5008	0.4931	0.5213	0.5128	0.5019	**0.5260**
	MCC	0.0	0.0132	0.0120	-0.0165	0.0628	0.0465	0.0439	**0.0640**
LinearSVC	macro F1	0.3324	0.3333	0.3367	0.4643	0.4306	0.3894	0.3376	**0.4390**
	micro F1	0.4981	0.5000	0.5000	0.4942	0.5092	0.5058	0.5019	**0.5212**
	MCC	-0.0439	0.0	0.0	-0.0131	0.0276	0.0237	0.0439	**0.0657**
AdaBoost	macro F1	0.3354	0.3325	0.3400	0.3529	**0.4860**	0.4504	0.3325	0.3612
	micro F1	0.5000	0.4981	0.5000	0.4942	0.4885	0.4962	0.4981	**0.5019**
	MCC	0.0034	-0.0439	0.0	-0.0324	-0.0233	-0.0094	-0.0439	**0.0092**

Table 2.6 Classification performance of classifiers on the Gisette data set.

Evaluation		Method							
Classifier	Metrics	Imbalanced	Random	SMOTE	MDO	NRAS	SWIM	BAGAN	GL-GAN
MLPClassifier	macro F1	0.4618	0.6051	0.6161	0.6317	0.5888		0.4462	**0.8552**
	micro F1	0.5685	0.6528	0.6604	0.6710	0.6426	-	0.5558	**0.8568**
	MCC	0.2423	0.4246	0.4371	0.4486	0.4066		0.2418	**0.7277**
LinearSVC	macro F1	0.6053	0.6115	0.6115	0.8616	0.6718		0.3521	**0.8636**
	micro F1	0.6529	0.6571	0.6571	0.8636	0.7011	-	0.5086	**0.8650**
	MCC	0.4248	0.4318	0.4318	**0.7482**	0.5018		0.0930	0.7457
AdaBoost	macro F1	0.5226	0.5874	0.5669	0.3361	0.5718		0.5949	**0.6271**
	micro F1	0.5993	0.6400	0.6271	0.4850	0.6300	-	0.6199	**0.6679**
	MCC	0.3320	0.4001	0.3817	-0.0935	0.3848		0.2770	**0.4476**

directly uses original imbalanced data sets without adding minority samples; 2) Random [30], i.e., random oversampling, which inflates minority class by duplicating existing minority samples; 3) SMOTE [14], which generates minority samples by performing linear interpolation operations between minority samples and their nearest neighbors; 4) MDO [2], which produces minority samples that have the same Mahalanobis distance from the considered class mean with existing minority samples; 5) NRAS [81], which performs a noise removal process on the minority class first and then constructs synthetic samples from the remaining samples; 6) SWIM [86], which utilizes the distribution information of majority class to generate minority samples located at the same Mahalanobis distance from the majority class; and 7) BAGAN [72] which takes random noise as input and produces synthetic samples to balance the imbalanced data set. We adopt the implementations of Random and SMOTE methods provided by literature [60] and of MDO and NRAS methods provided by literature [54] in all experiments with default settings. BAGAN is developed upon its public source code[4].

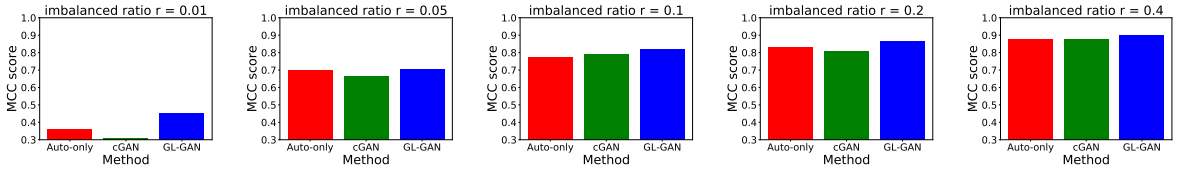
[4]https://github.com/IBM/BAGAN

For each imbalanced data set, we apply baselines and our model to generate synthetic minority data samples and then form different augmented data sets for training classifiers. Table 2.2 to Table 2.6 list the classification performance of three different classifiers on five test data sets. We conduct each experiment ten times and report average results. From these tables, we make the following observations: (i) Compared with the imbalanced set, the classification performance generally increases with the oversampling techniques, which shows the importance of oversampling. (ii) In most cases, with GL-GAN, the classification performance of classifiers outperforms with baselines, which implies the high quality of synthetic minority samples generated by GL-GAN. This is because local-based oversampling methods like SMOTE may produce some synthetic minority samples which are interleaved with existing majority samples or located in the null space of the given data set, while only global distribution information is explored in BAGAN and the generated synthetic samples may overlook the local structure of the given minority samples.

2.4.3 Components Analysis

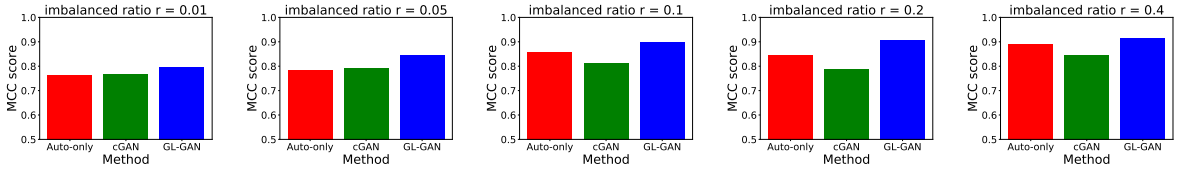
In order to investigate the impact of each module in our GL-GAN framework, we implement two models employing part of components contained in GL-GAN to generate synthetic minority samples and compare the quality of generated samples with GL-GAN. First, we combine the autoencoder component and the local data interpolator component together to obtain a new model called *Auto-only*. In Auto-only, the encoder E maps all given data samples into a latent space and the new synthetic minority samples are generated by the local data interpolator I. This procedure is the same with the first module in GL-GAN. However, instead of importing all latent representations into the generator G, Auto-only model employs the decoder Q to project all latent representations back to the raw feature space. Second, for studying the functionality of GAN-based module, we adopt conditional GAN [74] to generate synthetic minority samples. The conditional GAN takes random noise as input and produces synthetic samples with minority class label.

We conduct experiments on two real data sets and display the experimental results in Figure 2.3 and Figure 2.4, respectively. Here we can see, despite autoencoder (Auto-only) or conditional GAN (cGAN) can also produce synthetic minority samples for training a classifier, the quality



(a) $r = 0.01$ (b) $r = 0.05$ (c) $r = 0.1$ (d) $r = 0.2$ (e) $r = 0.4$

Figure 2.3 MCC score of AdaBoost on the Gisette data set.



(a) $r = 0.01$ (b) $r = 0.05$ (c) $r = 0.1$ (d) $r = 0.2$ (e) $r = 0.4$

Figure 2.4 MCC score of AdaBoost on the USPS data set.

of generated synthetic samples are not good enough, especially in the extremely imbalanced scenario ($r = 0.01$). However, since our GL-GAN could simultaneously explore the global and local information through combining the advantages of local-based oversampling techniques and generative adversarial learning together, the synthetic samples produced by GL-GAN could be more helpful for training a better classifier.

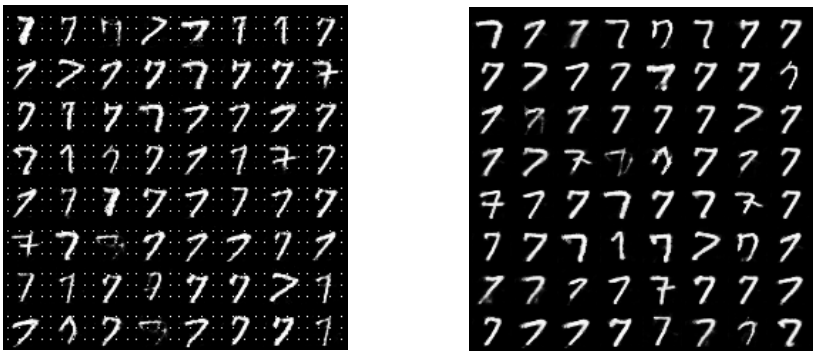
2.5 Case Study

For verifying whether GL-GAN can produce more realistic synthetic minority samples, we visualize the synthetic samples generated on a handwritten digits data set MNIST[5]. Here we randomly choose images “4" as majority class and images “7" as minority class, and form the imbalanced data set as described in Sec 2.4.1.1.

2.5.1 Functionality of Autoencoder

As we mentioned before, in order to avoid minority samples are generated in the null space of the given sample set or interleaved with majority samples, we require the encoder contained in our GL-GAN framework is able to map the given sample set $\mathcal{X}_{org}$ into two far-away clusters in the latent space, which can be achieved by Eq. (2.2). Here we utilize the MNIST data set to verify the usefulness of this design. In Figure 2.5, the right figure shows a snippet of images generated by the

[5]http://yann.lecun.com/exdb/mnist/



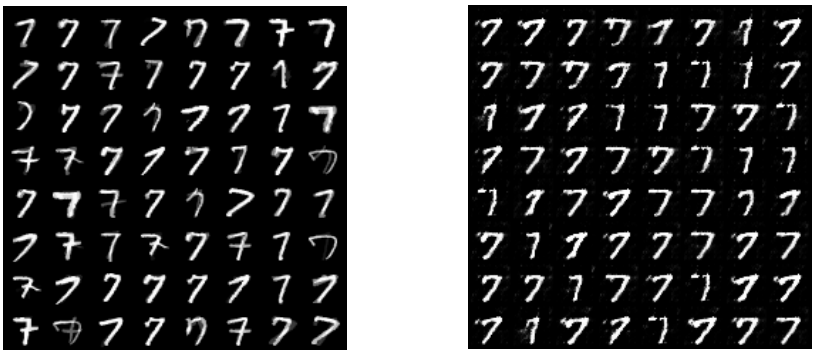
(a) Traditional AE (b) AE used in GL-GAN

Figure 2.5 Images generated by different autoencoders.

Auto-only method, in which the setting of the encoder is exactly same with the encoder E contained in our GL-GAN. As a comparison, we remove the loss function defined in Eq. (2.2) from the final loss function of the autoencoder, i.e., Eq. (2.3), and also apply SMOTE to generate synthetic minority samples in the latent space. In other words, the majority samples and minority samples are not required to be mapped far-away from each other in the latent space learned by the encoder, which is a common setting in the traditional autoencoder (AE). As the left figure shown, under this setting, the quality of generated synthetic minority samples “7" is worse than the Auto-only method generated ones. The reason is that, in the latent space, the synthetic minority samples generated by SMOTE may still have probability to interleave with majority samples if the majority cluster and minority cluster are not far-away from each other. Hence, the generated synthetic images may not good enough. In short, these two figures demonstrate the loss function described by Eq. (2.2) is useful and indispensable.

2.5.2 Quality of Generated Image Data

We also visualize the synthetic minority samples generated by SMOTE and our proposed GL-GAN framework on the MNIST data set. In the left figure of Figure 2.6, several synthetic samples produced by SMOTE looks like some intermediates between the majority class “4" and minority class “7". As we discussed before, due to only local neighbor relationships are utilized in SMOTE and the global information is totally ignored, SMOTE cannot avoid producing outliers or samples that interleaved with majority samples. On the contrary, our GL-GAN framework is able



(a) SMOTE (b) GL-GAN

Figure 2.6 Images generated by SMOTE and GL-GAN.

to generate more realistic synthetic minority samples. Since both global and local distributions are simultaneously explored in GL-GAN, the drawbacks of SMOTE can be overcame and high quality synthetic minority samples can be generated.

2.6 Chapter Conclusion

In this chapter, we propose a novel framework to solve the class imbalance problem through generating synthetic data samples for minority class. Different from local-based oversampling methods which only explore the local structure of minority samples and generative adversarial learning models which only utilize the global distribution information of all given samples, our GL-GAN framework considers both global and local information of the given data in the synthetic minority sample generation process. Extensive experimental results demonstrate that, comparing with existing baselines, our model can produce more realistic and discriminative synthetic minority samples, which are helpful for training better classifiers. In the future, we would extend our GL-GAN framework to the class imbalance problem of multi-class as well as some specific imbalanced application scenarios such as credit fraud detection.

CHAPTER 3

LEARNING FROM IMBALANCED CROWDSOURCED LABELED DATA

Crowdsourcing has proven to be a cost-effective way to meet the demands for labeled training data in supervised deep learning models. However, crowdsourced labels are often inconsistent and noisy due to cognitive and expertise differences among crowd workers. Existing approaches either infer latent true labels from noisy crowdsourced labels or learn a discriminative model directly from the crowdsourced labeled data, assuming the latent true label distribution is class-balanced. Unfortunately, in many real-world applications, the true label distribution typically is imbalanced across classes. Therefore, in this chapter, we address the problem of learning from crowdsourced labeled data with an imbalanced true label distribution. We propose a new framework, named "Learning from Imbalanced Crowdsourced Labeled Data" (ICED), which simultaneously infers true labels from imbalanced crowdsourced labeled data and achieves high accuracy on downstream tasks such as classification. The ICED framework consists of two modules, i.e., a true label inference module and a synthetic data generation module, that augment each other iteratively. Extensive experiments conducted on both synthetic and real-world data sets demonstrate the effectiveness of the ICED framework.

3.1 Chapter Introduction

The success of supervised deep learning models in many real-world applications, such as image classification [57, 89, 42] and speech recognition [33, 1, 38], is inseparable from the availability of large-scale labeled training data. However, obtaining a large amount of labeled data is often challenging. Annotating certain types of data samples such as medical images requires specific domain knowledge [93], while some other types of data such as videos or audios are expensive in terms of time [101]. By inviting multiple crowd workers to annotate labels for data samples simultaneously or sequentially, modern crowdsourcing platforms such as Amazon Mechanical Turk[1] offer a cost-effective way to collect large-scale labeled data [82]. Although crowdsourcing alleviates the label shortage problem to some extent, the annotated labels can be very inconsistent

[1] https://www.mturk.com

and noisy due to the cognitive differences between crowd workers [28]. For example, non-experts and experts may annotate the same object with distinct labels. As most existing supervised deep learning models only work well with determinate noise-free labels, there is a need for alternative approaches to handle such noisy labeled data.

In the past few decades, several approaches have addressed noisy crowdsourced labels. One class of approaches infer true labels from crowdsourced labels [22, 79, 104]. Another class of approaches learn a discriminative model directly from crowdsourced labeled data [82, 51, 92]. All the above approaches assume that the given training set is class-balanced, which is not true in real-world scenarios [95, 69], where *majority classes* have a significantly higher number of data samples than *minority classes*. Hence, those approaches perform poorly when training on imbalanced datasets.

There have been many attempts to address the challenges brought by imbalanced datasets, such as re-sampling approaches [67, 15, 35] and re-weighting approaches [21, 10]. These approaches require determinate noise-free training labels and are not able to handle data with crowdsourced labels. Therefore, there is a need for a new approach to address both imbalanced and noisy data in the crowdsourcing settings. To address this need, we study the problem of learning from imbalanced crowdsourced labeled data in this chapter. To the best of our knowledge, this is the first work to learn an effective discriminative model on crowdsourced labels when the latent true label distribution is imbalanced. Our goal is simultaneously obtaining accurate supervised information by inferring true labels from crowdsourced labels and ensuring good prediction performance of the classifier on all classes in the balanced test set.

In this chapter, we propose a novel framework ICED (Learning from Imbalanced Crowdsourced labEled Data). The ICED framework consists of two modules. One module uses generated synthetic data for minority classes to improve the true label inference process. Another module uses the inferred true labels to improve the quality of generated synthetic data. These two modules augment each other and improve themselves iteratively. After training, ICED can learn a classifier with good prediction performance on all classes uniformly distributed in the test set. The main contributions

of this chapter are summarized below:

- We are the first one to address the problem of learning from imbalanced crowdsourced labeled data, a more realistic scenario in the real world.
- We present a novel framework ICED, which can simultaneously infer true labels from imbalanced crowdsourced labeled data and achieve good prediction performance on all classes.
- We conduct extensive experiments on both synthetic and real datasets to demonstrate the effectiveness of ICED on the classification task.

3.2 The Proposed Framework

In this section, we first formulate the problem we studied in this chapter and then introduce our proposed ICED framework in detail.

3.2.1 Problem Formulation

Definition 3.2.1 (Learning from Imbalanced Crowdsourced Labeled Data)**.** Given a set of data $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, W crowd workers are invited to annotate every sample in $\mathbf{X}$ to produce a crowd label set $\mathbf{Y} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n\}$ and

$$\mathbf{y}_i = \{(y_{i,1}, w_{i,1}), (y_{i,2}, w_{i,2}) \ldots, (y_{i,W}, w_{i,W})\},$$

where each annotation pair $(y_{i,u}, w_{i,u})$ represents label $y_{i,u}$ provided by worker w_u for sample $\mathbf{x}_i$ from C classes, our goal is to obtain a deep neural network based classifier $\mathcal{F}$, which can achieve good prediction performance on uniformly distributed C-classes test data based on the data set $\mathbf{X}$ and corresponding crowdsourced label set $\mathbf{Y}$.

Note that, for each data sample $\mathbf{x}_i \in \mathbf{X}$, we assume it has W annotated labels. Moreover, as the true labels for sample data set $\mathbf{X}$ is unknown, we denote the estimated true labels inferred by our proposed framework for $\mathbf{X}$ as $\mathbf{T} = \{t_1, t_2, \ldots, t_n\}$. In this chapter, our focus is on the classification task for binary classes, i.e, there are two classes in the data set $\mathbf{X}$ and, one is the majority class and the other is the minority class. Note that our proposed ICED framework is also suitable for multi-class classification tasks with slight modifications and we will leave it as one future work.

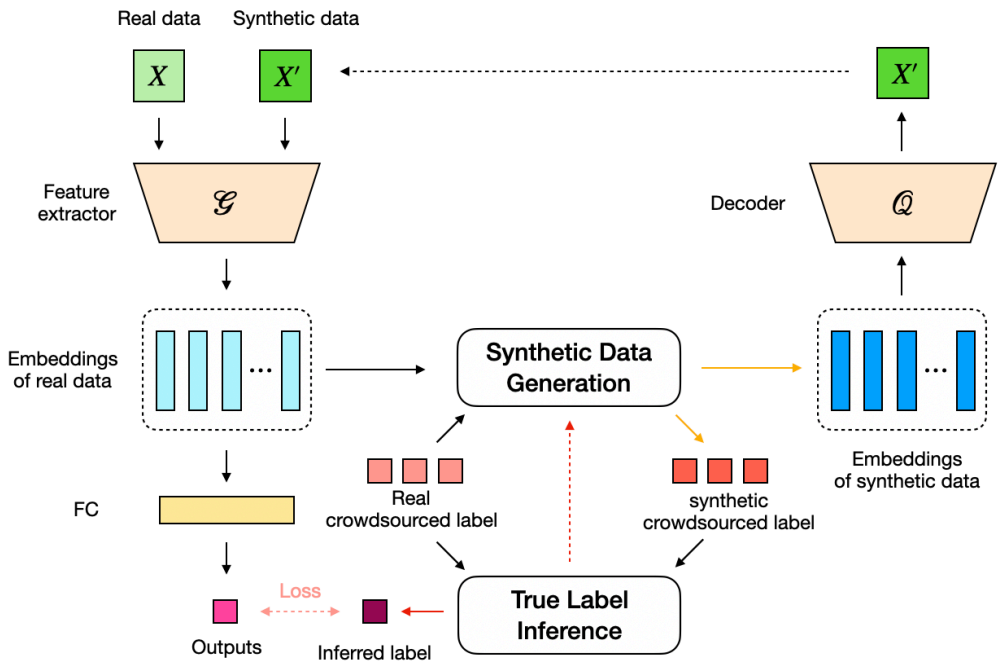


Figure 3.1 An overview of ICED. The solid yellow arrow and red arrow indicate outputs of synthetic data generation module and true label inference module in the current training iteration, respectively. The red dash arrow and black dash arrow represent the inferred labels obtained and synthetic data samples generated in the previous iteration, respectively.

3.2.2 Framework Overview

For tackling the learning from imbalanced crowdsourced labeled data problem, we propose a novel framework ICED as shown in Figure 3.1. The main structure of ICED is a deep neural network based classifier $\mathcal{F}$ consisting of a feature extractor $\mathcal{G}$ and a fully connected Layer (FC). During training the classifier $\mathcal{F}$, ICED introduces two modules: *true label inference* module and *synthetic data generation* module. The true label inference module estimates determinate true labels from given crowdsourced labeled data, and the synthetic data generation module generates synthetic data samples for minority class using the estimated true labels. These two modules augment each other and improve themselves iteratively. Furthermore, to make our ICED framework obtain better initial learning ability at the beginning of framework training phase, ICED also includes a warm-up training strategy specifically designed for the crowdsourced labeled data. Next, we introduce details of each component.

3.2.3 True Label Inference

Many classical approaches to infer true labels from crowdsourced labels ignore the correlation between data samples and cognitive differences between individual crowd workers. For example, some workers tend to judge class c_α as class c_β by mistake due to their cognitive differences. Therefore, for overcoming the aforementioned shortages, our ICED framework adopts an EM approach [22] into the true label inference module to estimate determinate labels from given crowdsourced labeled data.

To capture the annotation behaviors of crowd workers, we define $\Psi_{w_u}(c_\alpha, c_\beta)$ as the probability that worker w_u will annotate data samples with true label c_α as class c_β. Therefore, $\sum_{c_\beta \neq c_\alpha} \Psi_{w_u}(c_\alpha, c_\beta)$ represents the annotation error rate of the worker w_u when true label of samples are c_α. Let $\mathbb{T}$ be the random variable representing the true label of sample set $\mathbf{X}$ (similarly $\mathbb{T}_i$ for sample $\mathbf{x}_i$) and $\Phi_{c_\alpha} = p(\mathbb{T} = c_\alpha) = p(\mathbb{T}_i = c_\alpha)$ be the prior of class c_α, in the absence of any observations.

Our task is to estimate the probability of each label c_α ($c_\alpha \in [C]$) to be the latent true label for each sample $\mathbf{x}_i$ based on the crowdsourced labels $\mathbf{Y}$, i.e., $p(\mathbb{T}_i = c_\alpha | \mathbf{Y})$. The label with maximal probability is then chosen as the current estimated true label to train the deep neural network based classifier $\mathcal{F}$. The steps in the EM algorithm to estimate true labels are:

- E-step: computes the likelihood function of the observed crowdsourced labels $\mathbf{Y}$ based on current estimated true labels $\mathbf{T}$ and parameters $\Psi = \{\Psi_{w_u}(c_\alpha, c_\beta) | w_u \in [W], c_\alpha, c_\beta \in [C]\}$ and $\Phi = \{\Phi_{c_\alpha} | c_\alpha \in [C]\}$;
- M-step: updates the parameters by maximizing the likelihood function and refine the estimated true labels with new parameters.

In detail, we assume the labels provided by crowd workers are independently distributed. Given the current estimated true labels $\mathbf{T}$ and parameters Ψ and Φ, the likelihood of the observed crowdsourced

labels $\mathbf{Y}$ can be obtained by:

$$Q(\mathbf{Y}|\Psi, \Phi, \mathbf{T}) \propto \prod_{i\in[n]} p(\mathbb{T}_i = t_i) \prod_{u\in[W]} \Psi_{w_u}(t_i, y_{i,u}), \tag{3.1}$$

where i, u are the indices of data sample and crowd worker, respectively; $[n]$ and $[W]$ denote the sets $\{1, 2, \ldots, n\}$ and $\{1, 2, \ldots, W\}$, respectively.

The parameters in Ψ and Φ are updated by maximizing the above likelihood function. Specifically, Ψ can be computed as

$$\Psi_{w_u}(c_\alpha, c_\beta) = \frac{d(w_u, c_\alpha, c_\beta)}{d(w_u, c_\alpha)},$$

where $d(w_u, c_\alpha, c_\beta)$ represents the number of samples labeled as c_β by worker w_u when their current estimated true labels is c_α, and $d(w_u, c_\alpha)$ represents the number of samples labeled by worker w_u when their current estimated true labels is c_α. In addition, Φ can be computed as

$$\Phi_{c_\alpha} = \frac{\text{\# samples whose true label is estimated as } c_\alpha}{\text{\# samples in data set } \mathbf{X}}.$$

Based on these updates, we can refine the estimation of true label by Bayes's theorem

$$\begin{aligned} p(\mathbb{T}_i = c_\alpha|\mathbf{Y}, \Psi, \Phi) &\propto p(\mathbf{Y}|\Psi, \Phi, \mathbb{T}_i = c_\alpha) p(\mathbb{T}_i = c_\alpha) \\ &\propto p(\mathbb{T}_i = c_\alpha) \prod_{u\in[W]} \Psi_{w_u}(c_\alpha, y_{i,u}), \end{aligned} \tag{3.2}$$

and choose the label c_α with highest probability as the current estimated true label for data sample $\mathbf{x}_i$. We repeat E-step and M-step iteratively until convergence.

In summary, the true label inference module can provide two important information for our ICED framework: 1) an estimation of latent true labels $\mathbf{T}$, which can be used as supervised label information to train the deep neural network based classifier $\mathcal{F}$; 2) the marginal distribution $p(\mathbb{T} = c_\alpha)$ which reveals the data imbalance level between classes and thus guides the synthetic data generation module to augment a balanced synthetic data set. Moreover, we can also obtain a by-product from the EM algorithm, i.e., the annotation error rate of each worker w_u derived from $\Psi_{w_u}(\cdot, \cdot)$, which can be potentially used to eliminate or penalize the unqualified workers whose error rate is relatively high, depending on different application scenarios.

3.2.4 Synthetic Data Generation

The performance of the EM approach adopted in the true label inference module depends on the choice of prior probability, e.g., Φ_{c_α}, for initialization. Conventionally, uniform prior is used for initialization resulting in poor performance on imbalanced crowdsourced labeled data sets. Motivated by over-sampling approaches as an effective solution for an imbalanced data set, we integrate a synthetic data generation module in our ICED framework to balance the training data set.

As shown in Figure 3.1, we first apply the true label inference module to obtain estimated true labels $\mathbf{T}$. We then use the feature extractor $\mathcal{G}$ in the deep neural network based classifier $\mathcal{F}$ to map all data samples in $\mathbf{X}$ from the raw data space into a latent embedding space. Finally, we apply the following synthetic data generation process in the embedding space.

Suppose $\mathbf{z}$ be the embedding of the data sample $\mathbf{x}$ in the learned latent space, based on the information involved in the estimated true labels $\mathbf{T}$, all embeddings $\mathbf{z}_i$ which its corresponding determinate label t_i belongs to the minority class will be selected as candidate embeddings to help generate synthetic minority samples. After that, we utilize the linear interpolation operations adopted in the SMOTE [15] approach as the way to create synthetic minority sample embeddings. Specifically, for any candidate embedding $\mathbf{z}_i$, we (i) discover k nearest neighbors $\{\mathbf{z}_i^1, \mathbf{z}_i^2, \ldots, \mathbf{z}_i^k\}$ for $\mathbf{z}_i$; and (ii) randomly pick up one nearest neighbor $\mathbf{z}_i^r$ from the set $\{\mathbf{z}_i^1, \mathbf{z}_i^2, \ldots, \mathbf{z}_i^k\}$ to create a synthetic minority sample embedding $\mathbf{z}_i'$ as follows:

$$\mathbf{z}_i' = \mathbf{z}_i + \delta \left(\mathbf{z}_i^r - \mathbf{z}_i\right), \tag{3.3}$$

where δ is a scalar in range $[0, 1]$. The step (ii) can repeat R times, and, finally, $R \times m$ synthetic minority sample embeddings will be generated when executing the same process on all selected candidate embeddings with size m.

Since the true label inference module cannot guarantee 100% accuracy on estimating latent true labels from crowdsourced labels, the estimated determinate labels still have a chance to be opposite to the latent true labels. Hence, to reduce the adverse effects of possible wrong inference, different

from the SMOTE approach, which chooses δ randomly, we assign the value for δ based on the label certainty score of a sample $\mathbf{x}_i$ and its selected neighbor $\mathbf{x}_i^r$.

Definition 3.2.2 (Label Certainty Score)**.** Given a data example $\mathbf{x}_i$ and we assume that its crowdsourced labels $\{y_{i,u} \mid u \in [W]\}$ follow the multinomial distribution. The label certainty score $S(\mathbf{x}_i)$ is defined as the inverse variance of this distribution and is computed as:

$$S(\mathbf{x}_i) = \frac{1}{\mathbb{E}_u \|y_i - \mathbb{E}_u(y_i)\|^2 + \epsilon}, \tag{3.4}$$

where $\mathbb{E}_u$ is the expectation over crowdsourced label $\{y_{i,u} \mid u \in [W]\}$ for sample $\mathbf{x}_i$, and ϵ is a small constant to avoid numerical issue.

The label certainty score measures the agreement degree among crowd workers. Label certainty score reaches its minimum value when a tie or a draw happens and goes to its maximum value when all annotated labels for one data sample are consistent.

Therefore, given sample $\mathbf{x}_i$ and its neighbor $\mathbf{x}_i^r$, δ can be calculated by:

$$\delta = S(\mathbf{x}_i^r) / \big(S(\mathbf{x}_i) + S(\mathbf{x}_i^r)\big) + \eta, \tag{3.5}$$

where η is sampled from a uniform distribution to add some randomness on the scalar δ. With the help of Eq. (3.5), the generated synthetic embeddings $\mathbf{z}_i'$ will close to the candidate embedding which has larger label certainty score, such that the probability of the generated $\mathbf{z}_i'$ is located in the minority embedding clusters can be increased and, finally, the imbalanced issue can be alleviated via the aforementioned generation process.

After the synthetic minority sample embeddings are generated, we introduce the k-NN approach into the latent embedding space to construct synthetic crowdsourced labels for generated sample embeddings. More specifically, for any generated minority embedding $\mathbf{z}_i'$, we collect crowdsourced labels of its k nearest neighbor embeddings of real data samples and then determine its synthetic crowdsourced labels by simulating the annotation behavior of each crowd worker in these collected k crowdsourced labels.

Algorithm 3.1 The algorithm of warm-up training.

Input: sample set $\mathbf{X}$, crowdsourced label set $\mathbf{Y}$

Calculate label agreement score $S(\mathbf{x}_i)$ for each sample $\mathbf{x}_i \in \mathbf{X}$.
Obtain estimated true label t_i for each $\mathbf{y}_i \in \mathbf{Y}$ using MV.
Random initialize parameters of classifier $\mathcal{F}$.
Divide sample set $\mathbf{X}$ and estimated true label set $\mathbf{T}$ into four different groups based on their label certainty scores.
for warm-up epochs **do**
    Train the classifier $\mathcal{F}$ using the data samples and determinate labels in the third highest certainty group.
end for
for warm-up epochs **do**
    Train the classifier $\mathcal{F}$ using the data samples and determinate labels in the second highest certainty group.
end for
for warm-up epochs **do**
    Train the classifier $\mathcal{F}$ using the data samples and determinate labels in the highest certainty group.
end for

When we obtain synthetic minority sample embeddings and corresponding synthetic crowdsourced labels, as shown in Figure 3.1, we use a pre-trained decoder Q to map the synthetic embeddings back to the raw data space and will use the augmented balanced training set to update the parameters of the deep neural network based classifier $\mathcal{F}$.

In summary, the synthetic data generation module in our ICED framework addresses the issues causes by imbalanced training data set via generating sufficient synthetic minority samples with synthetic crowdsourced labels, which can benefit both the true label inference process and the deep neural network training.

3.2.5 Warm-up Training

Recent studies have discovered that deep neural networks can learn even on noisy labeled data [76, 61]. Hence, a warm-up training phase is an effective strategy to initialize supervised deep learning models. Existing literature [59, 36] uses all available data in the warm-up training phase. Different from existing literature, in our ICED framework, we design a new warm-up training strategy specifically designed for the crowdsourced labeled data.

As shown in Algorithm 3.1, given data set $\mathbf{X}$ and crowdsourced label set $\mathbf{Y}$, we first calculate

Algorithm 3.2 The algorithm of ICED.

Input: sample set $\mathbf{X}$, crowdsourced label set $\mathbf{Y}$

1: Conduct warm-up training as described in Algorithm 3.1.
2: **repeat**
3: Generate synthetic minority samples and corresponding synthetic crowdsourced labels as described in Sec. 3.2.4.
4: Obtain the inferred true label set $\mathbf{T}'$ for the augmented crowdsourced labels as described in Sec. 3.2.3.
5: Train the classifier $\mathcal{F}$ using the augmented data samples and inferred determinate labels $\mathbf{T}'$.
6: **until** model converge or maximum training epoch reached

the label certainty score for each data sample $\mathbf{x}_i$. After gathering label certainty scores for all data samples, we apply majority voting (MV) on the crowdsourced label set $\mathbf{Y}$ to obtain an estimated true label set $\mathbf{T}$. Each element t_i in $\mathbf{T}$ is obtained by aggregating the corresponding crowdsourced label $\mathbf{Y}_i$ using MV. We then divide the sample set $\mathbf{X}$ and corresponding true label set $\mathbf{T}$ into four different subgroups based on the label certainty scores: low certainty group, third-highest certainty group, second highest certainty group, and highest certainty group. We use all data samples except those in the low certainty group to initially train the deep neural network based classifier $\mathcal{F}$ with associated determinate labels in a supervised way.

In general, there is a higher probability of the determinate label t_i, obtained by MV, being the same as the latent true label when sample $\mathbf{x}_i$ has a higher label certainty score $S(\mathbf{x}_i)$. Our new warm-up training strategy is similar to using noisy labeled data to help provide the initial ability for the deep neural network based classifier $\mathcal{F}$ and using clean labeled data to fine-tune $\mathcal{F}$. After the warm-up training phase, the ICED framework can get a better initial prediction ability.

3.2.6 Algorithm

In this subsection, we present our ICED framework for learning from imbalanced crowdsourced labeled data in Algorithm 3.2.

As shown in Algorithm 3.2, we first introduce our designed warm-up training strategy to make the deep neural network based classifier $\mathcal{F}$ obtain better initial ability. Then, in each training epoch, we apply the synthetic data generation module to produce synthetic minority samples with synthetic crowdsourced labels for balancing the training data set. After that, the true label inference module is

used to inferred latent true labels for the augmented crowdsourced labels. Hence, the parameters of the classifier $\mathcal{F}$ can be updated based on the augmented balanced data samples and corresponding inferred determinate labels in a supervised way. We continuously conduct this iteration process until $\mathcal{F}$ converges or the maximum training epoch is reached.

3.3 Experiment

In this section, we conduct experiments to verify the effectiveness of our proposed ICED framework by answering the following three questions:

1. Can the proposed framework obtain good prediction performance on the balanced test data?
2. Does the generated synthetic data improve the accuracy of the true label inference process?
3. Does our newly designed warm-up training strategy improve over existing warm-up training strategies?

To answer the first question, we compare the performance of ICED with several state-of-the-art crowdsourced label processing approaches on the classification task. For the second question, we compare the accuracy of true label inference with and without synthetic data generation modules on two synthetic datasets. Finally, we compare the prediction performance of the deep neural network based classifier $\mathcal{F}$ using our designed warm-up training strategy and by traditional warm-up training strategies to answer the third question.

3.3.1 Data Sets

3.3.1.1 Synthetic Data Sets

We conduct experiments on three synthetic data sets and one real-world data set. Table 3.1 summarizes key statistical information of these four data sets. The three synthetic imbalanced crowdsourced labeled data sets are constructed based on three widely used data sets: Gisette, USPS, and Gas Sensor Array Drift (GSAD). Specifically, Gisette and USPS datasets are from Feature Selection data repository[2] and the GSAD data set is from UCI data repository[3]. Next,

[2]http://featureselection.asu.edu/datasets.php
[3]https://archive.ics.uci.edu/ml/index.php

Table 3.1 Statistics of data sets. The entries in "# majority class" and "# minority class" represent the number of samples we used for those classes, respectively, to construct a synthetic training data set.

Statistic item	Dataset			
	Gisette-Syn	USPS-Syn	GSAD-Syn	Emotion
# features	5,000	256	128	1,582
# training data	3,080	734	2,575	3,027
# majority class	2,800	668	2,341	-
# minority class	280	66	234	-
# crowd worker	7	9	11	5
# test data	1,400	332	1,170	900

using the USPS data set as an illustrative example, we describe how we construct a synthetic imbalanced crowdwourced labeled data set *USPS-Syn*. First we randomly choose one class as a majority class and another one as a minority class from the total ten classes contained in the USPS data set to obtain a balanced binary data set. Then we split 80% data samples in this balanced binary data set as candidate training set and the rest as the test set. Note that the test set is class-balanced. Different from previous crowdsourced label processing approaches that randomly assign a mislabeling probability for each worker to all data samples [3, 50], in this chapter, we present a new way to synthesize the crowdsourced labels by considering the difficulties of data samples. Intuitively, it should be easier to infer true labels from data samples with a higher label certainty score (e.g., all W crowd workers annotate the same label for the same data sample). Motivated by this intuition, we introduce a deep neural network to evaluate the identification difficulty of each sample. Specifically, we train a deep neural network based on the candidate training set and stop the training process when the training accuracy is higher than 98%. Then, for each class, we use the softmax outputs of the trained deep neural network to indicate the identification difficulties of data samples. Then, we assign a mislabeling probability for each data sample based on the relative ease of inferring from that data sample. Higher the difficulty in inferring from a data sample, the higher the mislabeling probability for all crowd workers. Finally, we remove 90% minority samples from the candidate training set based on their ground truth labels to obtain an imbalanced crowdsourced labeled data set USPS-Syn.

Another two synthetic data sets Gisette-Syn and GSAD-Syn can be constructed in the similar way as mentioned above. Once again, we textitasize that only the training set in these three synthetic data sets is an imbalanced crowdsourced labeled data set while the test set is a class-balanced data set with determinate labels.

3.3.1.2 Real Data Set

We collected a real-world imbalanced crowdsourced labeled data set *Emotion* from our educational practice. The collected data samples in the Emotion dataset are 1-minute audio tracks collected from multiple teachers who teach courses such as Mathematics and English in primary school. We split all audio tracks in Emotion into a training set and a test set with sample size 3,027 and 900 separately. Five teaching professionals are invited to annotate every audio track in the training set as either high emotion arousal or low emotion arousal to assess teaching effects on courses and the annotation results provided by one teaching expert for audio tracks in the test set are adopted as the ground truth labels. As the original data samples in the Emotion data set are audio tracks, neither our ICED framework nor baseline methods can directly deal with those data samples. For addressing this issue, we apply OpenSmile[4] to extract 1,582 acoustic features, such as signal energy, loudness, MFCC features, etc., from the collected audio tracks. Since the Emotion dataset is collected from our educational practice, it cannot guarantee the latent true label distribution in the training set is class-balanced. Moreover, based on the label inference results produced by majority voting, among the total of 3,027 samples in the training set, there are 1,911 samples in one class and 1,116 samples in the other class. For experiment purpose, we maintained the same number of data samples in each class in the test set.

3.3.2 Performance Comparison

3.3.2.1 Baseline Methods

For evaluating the effectiveness of our proposed ICED framework on the learning from imbalanced crowdsourced labeled data problem, we compare the performance of ICED with several representative state-of-the-art crowdsourced label processing approaches on the classification task,

[4]https://www.audeering.com/opensmile/

including:

- Majority Voting (MV), which infers determinate labels based on the majority of annotated labels.
- D&S [22], which infers determinate labels via estimating the error rate of each crowd worker.
- Crowd-Layer [82], which is an end-to-end deep neural network containing a novel crowd layer to learn from crowdsourced labeled data directly.
- MBEM [51], which is able to learn from crowdsourced labeled data via jointly modeling latent true labels and crowd worker qualifications.
- CPC [48], which improves the performance of classifier via learning parameters of classifier and clusters of crowd workers jointly.

As MV and D&S can only infer determinate labels instead of learning a classifier from crowdsourced labels, we introduce two classifiers Logistic regression (LR) and deep neural networks (DNN). Specifically, we train LR and DNN on the same datasets with determinate labels inferred by MV and D&S individually and use them as baseline methods. We denote these baseline methods as *MV+LR*, *MV+DNN*, *D&S+LR* and *D&S+DNN*.

Table 3.2 shows the classification performance of our ICED framework by comparing against seven baseline methods on three synthetic data sets and one real data set. Based on this table, we have the following observations. First, the classification performance of both LR and DNN, measured in terms of accuracy and F1-score, is higher when using MV instead of D&S to infer determinate labels. D&S, as an EM-based approach, assumes a uniform label distribution. MV independently aggregates annotated labels of each crowdsourced label. Hence, given an imbalanced crowdsourced labeled data set, the performance of MV on the true label inference task will not be affected by the imbalanced true label distribution. On the contrary, D&S may show poor performance due to its inaccurate uniformity assumption. Second, our ICED framework achieves

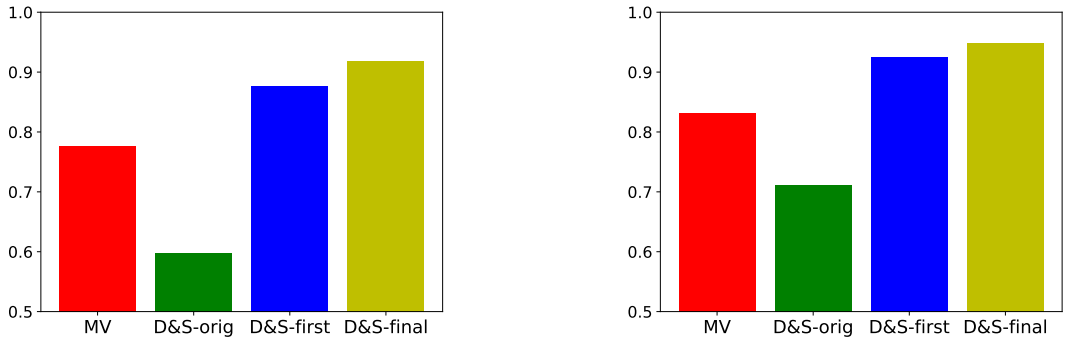
Table 3.2 Classification performance of our ICED framework and baseline methods on four data sets.

	Gisette-Syn		USPS-Syn		GSAD-Syn		Emotion	
Methods	Accuracy	F1-score	Accuracy	F1-score	Accuracy	F1-score	Accuracy	F1-score
MV+LR	0.8179	0.8175	0.8494	0.8480	0.7094	0.6872	0.8289	0.8277
MV+DNN	0.8000	0.7979	0.8735	0.8732	0.8333	0.8327	0.7944	0.7901
D&S+LR	0.7671	0.7592	0.8193	0.8136	0.6974	0.6716	0.8311	0.8294
D&S+DNN	0.7636	0.7562	0.7530	0.7386	0.8085	0.8082	0.7811	0.7749
Crowd-Layer	0.8200	0.8167	0.9006	0.8998	0.8342	0.8295	0.8300	0.8273
MBEM	0.6967	0.4211	0.7813	0.5334	0.6826	0.5577	0.6344	0.5324
CPC	0.8021	0.8020	0.8313	0.8311	0.6154	0.5917	-	-
ICED	**0.8521**	**0.8512**	**0.9036**	**0.9030**	**0.8872**	**0.8865**	**0.8644**	**0.8640**

the best classification performance on all four data sets comparing with several representative state-of-the-art crowdsourced label processing approaches. We believe there are three reasons behind this phenomenon. First, even though the D&S approach assumes uniformity in data distribution, ICED generates synthetic data to augment the imbalances between classes in the training set. The resulting training set will approximate a uniform distribution, enhancing the performance of the D&S approach. Second, the more accurate determinate labels inferred by the true label inference module improves the synthetic data generation module. The reason being, the synthetic data generation module can use the inferred determinate labels to differentiate minority data samples from majority ones to generate synthetic samples in minority classes. As a result, the data samples produced by the synthetic data generation module have a higher probability of belonging to the minority classes. Third, the synthetic generated data can also help the classifier $\mathcal{F}$ in ICED to obtain better generalization ability during the model training phase via augmenting the imbalanced training set. In summary, comparing with several representative state-of-the-art crowdsourced label processing approaches, our ICED framework is more effective to tackle the problem of learning from imbalanced crowdsourced labeled data.

3.3.3 Ablation Study

As we mentioned before, the D&S approach assumes uniform label distribution as prior knowledge for initialization. Therefore, the true label inference performance of the D&S approach is lower than the MV approach on the imbalanced crowdsourced labeled dataset. The ICED frame-



(a) Gisette-Syn.

(b) GSAD-Syn.

Figure 3.2 Accuracy of true label inference using MV and the true label inference module (D&S) in ICED.

work addresses the issue in the D&S approach by integrating a synthetic data generation module. The synthetic data generation module balances the imbalanced training set via generating synthetic data samples for minority classes. The resulting augmented dataset better fits the prior knowledge used in D&S.

To verify whether and how the synthetic generation module benefits from the true label inference module in our ICED framework, we compare the true label inference accuracy of D&S adopted in ICED with MV. We show the comparison on two synthetic datasets— Gisette-Syn and GSAD-Syn— because the ground truth labels are available for these two datasets. In our experiments, we record the true label inference accuracy of D&S for three cases: 1) before introducing the synthetic data generation module, 2) after applying the synthetic data generation module once, and 3) after completing the training procedure of ICED. We denote these three cases as *D&S-orig*, *D&S-first*, *D&S-final*, respectively. In Figure 3.2, we find that the performance of D&S varies widely for different cases. Take the experimental results obtained on the training set of Gisette-Syn as an example. As shown in Figure 3.2a, before introducing the synthetic data generation module, the true label inference accuracy of D&S is below 60%, which is much worse than MV. Surprisingly, by conducting the synthetic data generation process just once, the label inference accuracy of D&S is higher than 80%. After finishing the training procedure of ICED, i.e, after repeating the synthetic data generation process multiple times, D&S achieves higher than 90% true

Table 3.3 Performance of different warm-up strategies.

Datasets	Methods	# samples	# epochs	Accuracy	F1-score
Gisette-Syn	Trad-I	3,080	15	0.8014	0.7989
	Trad-II	2,043	15	0.6079	0.5372
	ICED-w	2,043	5×3	**0.8186**	**0.8126**
USPS-Syn	Trad-I	734	6	0.7500	0.7333
	Trad-II	103	6	0.7922	0.7828
	ICED-w	103	2×3	**0.8373**	**0.8329**
GSAD-Syn	Trad-I	2,575	6	0.4581	0.3142
	Trad-II	507	6	0.4504	0.3105
	ICED-w	507	2×3	**0.8376**	**0.8332**

label inference accuracy, which is a significant improvement in comparison to a naive application of D&S on the imbalanced crowdsourced labeled dataset. In conclusion, the synthetic generation module significantly enhances the performance of the true label inference module in ICED.

3.3.4 Effectiveness of Warm-up Training

In this subsection, we test the effectiveness of our designed warm-up training strategy. Given a set of crowdsourced labeled data, the warm-up training strategy adopted in our ICED framework first calculates label certainty score for each data sample based on its corresponding crowdsourced label. Then it divides data samples into different groups based on their label certainty scores. Data samples in the third-highest certainty group will feed the classifier $\mathcal{F}$ in ICED first with their corresponding determinate labels produced by MV. Data samples in the highest certainty group will train $\mathcal{F}$ after those in the second-highest group are picked. In experiments, we denote our designed warm-up training strategy as *ICED-w*. As a comparison, we implement one common warm-up training strategy used in literature for learning from noisy labeled data that uses all available data simultaneously to warm up the model. We denote this warm-up strategy as *Trad-I*. Another warm-up training strategy *Trad-II*, which is the same as Trad-I, except it only uses data samples in the highest, second-highest, and third-highest certainty groups rather than all the available data samples. In other words, Trad-II chooses the same data samples adopted in our designed warm-up training strategy ICED-w and uses them to feed $\mathcal{F}$ at the same time. For evaluation, we report the classification performance of $\mathcal{F}$ by training on different warm-up training strategies in Table 3.3.

We observe that the classifier $\mathcal{F}$ training by ICED-w achieves the best classification performance, comparing with Trad-I and Trad-II, on all datasets. Thus, our designed warm-up training strategy more effectively initializes ICED.

3.4 Related Work

3.4.1 Processing Crowdsourced Labels

Inferring true labels from crowdsourced labels is a challenge as the crowd workers have diverse expertise [113]. A naive approach to infer true labels is majority voting (MV), which uses the majority of annotated labels as the true label. The MV approach performs poorly in practice, as the crowd workers have diverse expertise and reliability. An Expectation-Maximization (EM) [22] approach addresses the differences between crowd workers by estimating the error rate of each crowd worker from the crowd labels. Therefore, an EM approach has higher accuracy than MV in inferring true labels. Inspired by this, Whitehill et al. [104] used an iterative approach considering both sample difficulty and crowd worker reliability to infer true labels. The above approaches focus only on inferring true labels. Some recent works integrate true labels inference with downstream tasks. Kajino et al. [48] developed a clustered personal classifier method that simultaneously trains a classifier and estimates a cluster of workers. Rodrigues et al. [83] generalized Gaussian process classification considering crowd workers with diverse expertise. Raykar et al. [79] designed an EM-based approach to jointly learn a crowd worker noise model and a regression model. Khetan et al. [51] proposed another EM-based approach for learning from crowdsourced labeled data by jointly modeling latent true labels and crowd worker qualifications. Guan et al. [34] modeled information from each worker and then learned combination weights via back-propagation. As all the above approaches assume a uniformed label distribution as prior knowledge for initialization, they cannot achieve good generalization when the given training set has an imbalanced true label distribution.

3.4.2 Handling Imbalanced Data

The performance of a classifier heavily relies on the quality and quantity of training data [49]. Since the majority of classes in the imbalanced training set can dominate the loss function of training, classifiers trained on imbalanced data often generalize poorly. Existing approaches to handle imbalanced data mainly falls into two categories: re-sampling and re-weighting. Re-sampling approaches balance the imbalanced data through under-sampling data samples from majority classes [110, 67] or over-sampling data samples from minority classes [15, 100]. As under-sampling approaches often discard several data samples, over-sampling approaches are better in practice. Synthetic Minority Over-sampling Technique (SMOTE) [15] is a well-accepted over-sampling approach. Instead of duplicating existing minority data samples to inflate minority classes, SMOTE produces unseen synthetic minority samples by applying linear interpolation operations between a specific minority sample and one of its nearest neighbors within the same class. Several variants of SMOTE [35, 39] further improve the prediction performance of classifiers training on imbalanced datasets. Re-weighting approaches allocate different weights for different classes or even different data samples. For example, Lin et al. [63] proposed Focal loss to reshape the standard cross entropy loss such that it down-weights the loss assigned to well-classified data samples. Cui et al. [21] presented to utilize the data overlap measurement to quantify the effective number of samples for each class and re-weight each class by the inverse of the number of effective samples per class. Existing imbalanced data handling approaches assume that the given labels are determinate and noise-free, which is not the case in crowdsourcing settings. Therefore, learning from imbalanced crowdsourced labels needs to be addressed.

3.5 Chapter Conclusion

In this chapter, we investigate the problem of learning from imbalanced crowdsourced labeled data. We present a novel ICED framework to deal with the imbalanced true label distribution and noisy crowdsourced labels. The ICED framework alleviates the negative impacts of imbalanced true label distribution while using the supervised information in the crowdsourced labels. To evaluate the performance of the ICED framework, we apply ICED into a classification task by

training on both synthetic and real imbalanced crowdsourced labeled datasets and comparing its performance with several representative crowdsourced label processing approaches. Extensive experimental results demonstrate the effectiveness of our proposed framework ICED on learning from imbalanced crowdsourced labeled data.

CHAPTER 4

IMBALANCED ADVERSARIAL TRAINING WITH REWEIGHTING

Adversarial training has been empirically proven to be one of the most effective and reliable defense methods against adversarial attacks. However, the majority of existing studies are focused on balanced data sets, where each class has a similar amount of training examples. Research on adversarial training with imbalanced training data sets is rather limited. As the initial effort to investigate this problem, we reveal the facts that adversarially trained models present two distinguished behaviors from naturally trained models in imbalanced data sets: (1) Compared to natural training, adversarially trained models can suffer much worse performance on under-represented classes, when the training data set is extremely imbalanced. (2) Traditional reweighting strategies which assign large weights to under-represented classes will drastically hurt the model's performance on well-represented classes. In this chapter, to further understand our observations, we theoretically show that the poor data separability is one key reason causing this strong tension between under-represented and well-represented classes. Motivated by this finding, we propose the Separable Reweighted Adversarial Training (SRAT) framework to facilitate adversarial training under imbalanced scenarios, by learning more separable features for different classes. Extensive experiments on various data sets verify the effectiveness of the proposed framework.

4.1 Chapter Introduction

The existence of adversarial samples [91, 32] has risen huge concerns on applying deep neural network (DNN) models into security-critical applications, such as autonomous driving [17] and video surveillance systems [58]. As countermeasures against adversarial attacks, adversarial training [71, 111, 103] has been empirically proven to be one of the most effective and reliable defense methods. In general, it can be formulated to minimize the model's average error on adversarially perturbed input examples [71]. Although promising to improve the model's robustness, most existing adversarial training methods assume that the number of training examples from each class is equally distributed. However, datasets collected from real-world applications typically have imbalanced distribution [27, 64]. Hence, it is natural to ask: *What is the behavior of adversarial training*

under imbalanced scenarios? Can we directly apply existing imbalanced learning strategies in natural training to tackle the imbalance issue for adversarial training? Recent studies find that adversarial training usually presents distinct properties from natural training. For example, compared to natural training, adversarially trained models suffer more from the overfitting issue [85], and they tend to present strong class-wise performance disparities, even if the training examples are uniformly distributed over different classes [108]. Imagine that if the training data distribution is highly imbalanced, these properties of adversarial training can be greatly exaggerated and make it extremely difficult to be applied in practice. Therefore, it is necessary but challenging to answer aforementioned questions.

As the initial effort to study the imbalanced problem in adversarial training, in this work, we first investigate the performance of existing adversarial training under imbalanced settings. As a preliminary study shown in Section 4.2.1, we apply both natural training and PGD adversarial training [71] on multiple imbalanced training datasets constructed from CIFAR10 training dataset [56] and evaluate trained models' performance on class-balanced test dataset. From the preliminary results, we observe that, compared to naturally trained models, adversarially trained models always present very low standard & robust accuracy[1] on under-represented classes. This observation suggests that adversarial training is more sensitive to imbalanced data distribution than natural training. Thus, when applying adversarial training in practice, imbalance learning strategies should be considered for help.

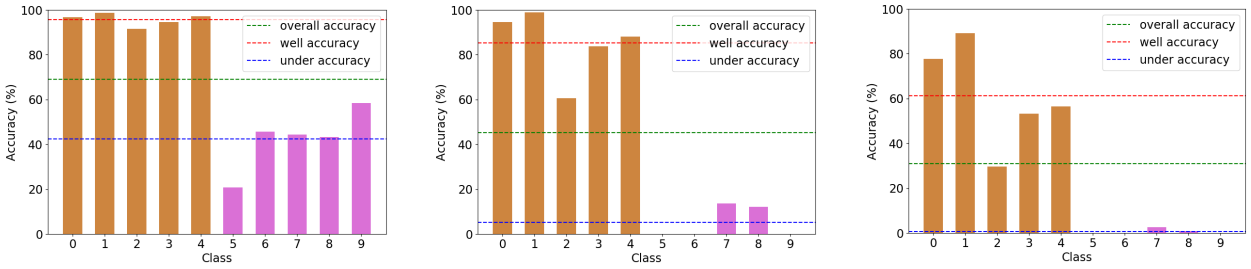
As a result, we explore potential solutions which can handle the imbalance issue for adversarial training. In this chapter, we focus on studying the behavior of the *reweighting* strategy [41] and leave other strategies such as resampling [26] for one future work. In Section 4.2.2, we apply the reweighting strategy to adversarial training with varied weights assigning to one under-represented class and evaluate trained models' performance. From the results, we observe that, in adversarial training, increasing weights for an under-represented class can substantially improve the standard & robust accuracy on this class, but drastically hurt the model's performance on the well-represented

[1] In this chapter, we denote *standard/robust accuracy* as model's accuracy on the input examples without/with perturbations, respectively. Without clear clarification, we consider the perturbation is constrained by l_∞-norm 8/255.

class. This finding indicates that the performance of adversarially trained models is very sensitive to the reweighting manipulations and it could be very hard to figure out an eligible reweighting strategy which is optimal for all classes.

It is also worth noting that, in natural training, we find that upweighting the under-represented class increases model's standard accuracy on this class but only slightly hurts the accuracy on the well-represented class, even when adopting a large weight for the under-represent class. To further investigate the possible reasons leading to different behaviors of the reweighing strategy in natural and adversarial training, we visualize their learned features (in Figure 4.3), and observe that features learned by the adversarially trained model of different classes tend to mix together while they are well separated for the naturally trained model. This observation motivates us to theoretically show that when the given data distribution has poor data separability, upweighting under-represented classes will hurt the model's performance on well-represented classes. Motivated by our theoretical understanding, we propose a novel framework SRAT (Separable Reweighted Adversarial Training) to facilitate the reweighting strategy in imbalanced adversarial training by enhancing the separability of learned features. Through experiments, we validate the effectiveness of SRAT. The main contributions of this chapter include:

- We empirically discover two major differences between naturally trained models and adversarial trained models under imbalanced settings, which reveal a fact that adversarial training alone cannot work well given an imbalanced training dataset.
- We theoretically verify the poor data separability is one key reason causing the failure of adversarial training based methods under imbalanced settings.
- We propose a novel framework SRAT to facilitate the reweighting strategy in imbalanced adversarial training and demonstrate the effectiveness of SRAT via extensive experiments.



(a) Natural Training Standard Acc. (b) Adv. Training Standard Acc. (c) Adv. Training Robust Acc.

Figure 4.1 Class-wise performance of natural & adversarial training using an imbalanced CIFAR10.

4.2 Preliminary Study

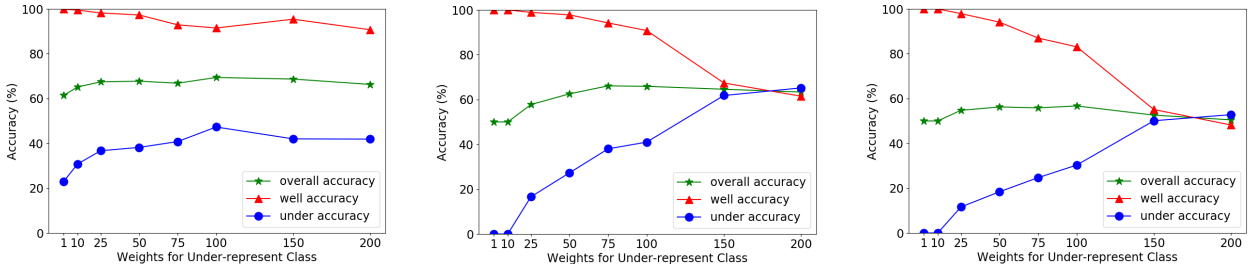
4.2.1 The Behavior of Adversarial Training

In this subsection, we conduct preliminary studies to examine the performance of PGD adversarial training [71]. Following previous works [21, 10], we construct an imbalanced CIFAR10 [56] training dataset, where each of the first 5 classes (a.k.a. well-represented classes) has 5,000 training examples and each of the last 5 classes (a.k.a. under-represented classes) has 50 training examples.

Figure 4.1 shows the performance of naturally and adversarially trained models using a ResNet18 [42] architecture. From the figure, we can observe that, compared with natural training, PGD adversarial training will result in a larger performance gap between well-represented classes and under-represented classes. For example, in natural training, the ratio between the average standard accuracy of well-represented classes (brown) and under-represented classes (violet) is about 2:1, while in adversarial training, this ratio expands to 16:1. Moreover, for adversarial training, it has extremely poor performance on under-represented classes. There are 3 out of the 5 under-represented classes with 0% standard & robust accuracy. As a conclusion, the performance of adversarial training is easier to be affected by imbalanced distribution than natural training and suffers more on under-represented classes. We also conduct more experiments under various imbalanced settings and get have similar findings.

4.2.2 The Reweighting Strategy in Natural Training v.s. in Adversarial Training

The preliminary study in Section 4.2.1 demonstrates that it is highly demanding to adjust the original adversarial training methods to accommodate imbalanced data distribution. Next, we

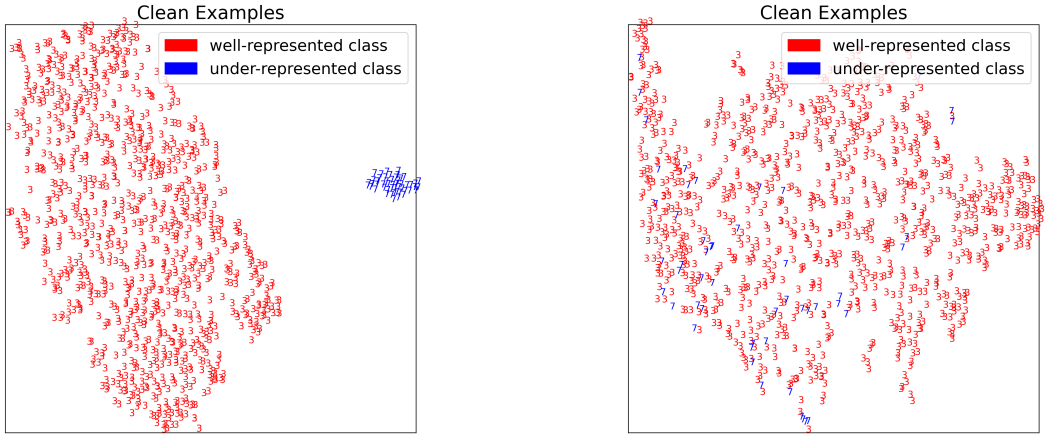


(a) Natural Training Standard Acc. (b) Adv. Training Standard Acc. (c) Adv. Training Robust Acc.

Figure 4.2 Class-wise performance of reweighted natural & adversarial training in binary classification.

investigate the effectiveness of adopting the reweighting strategy [41] in adversarial training. Our experiments are conducted under a binary classification setting, where the training dataset contains two classes that are randomly selected from CIFAR10 dataset, with each class having 5,000 and 50 training examples respectively. Based on this training dataset, we arrange multiple trails of (reweighted) natural training and (reweighted) adversarial training, with the weight ratio between the under-represented class and well-represented class ranging from 1:1 to 200:1.

Figure 4.2 shows the experimental results with training data sampled from the classes "cat" and "horse". As demonstrated in Figure 4.2, increasing the weight for the under-represented class (horse) will drastically increase the model's performance on this class, while also immensely decreasing the performance on the well-represented class (cat). For example, when increasing the weight ratio from 1:1 to 150:1, the standard accuracy of the under-represented class is improved from 0% to $\sim$ 60% and its robust accuracy from 0% to $\sim$ 50%. However, the standard accuracy on the well-represented class drops from 100% to 60%, and its robust accuracy drops from 100% to 50%. These results illustrate that adversarial training's performance can be significantly affected by the reweighting strategy. As a result, the reweighting strategy in this setting can hardly help improve the overall performance no matter which weight ratio is chosen, because the model's performance always presents a strong tension between these two classes. We also conduct more experiments using different binary imbalanced datasets and get have similar observations.



(a) Natural Training. (b) Adversarial Training.

Figure 4.3 t-SNE visualization of learned features.

4.3 Theoretical Analysis

In Section 4.2.2, we observe that in natural training, the reweighting strategy can only make a small impact on the two classes' performance. This phenomenon has been extensively studied by recent works [9, 107], where they find that a linear classifier optimized by SGD on a linearly separable data will converge to the solution of the *hard-margin support vector machine* [77]. In other words, as long as the data can be well separated, reweighting will not make huge influence on the finally trained models.

Inspired by their conclusions, we hypothesize that, as the adversarially trained models separate the data poorly, their performance is highly sensitive to the reweighting strategy. As a direct validation of our hypothesis, in Figure 4.3, we visualize the learned (penultimate layer) features of the imbalanced training examples used in the binary classification problem in Section 4.2.2. We find that adversarially trained models do present obviously poorer separability on the learned features. Next, we theoretically analyze the impact of reweighting on linear models which are optimized under poorly separable data.

Binary Classification Problem. To construct the theoretical study, we focus on a binary

classification problem, with a Gaussian mixture distribution $\mathcal{D}$ which is defined as:

$$y \sim \{-1, +1\}, \quad x \sim \begin{cases} \mathcal{N}(\mu, \sigma^2 I), & \text{if } y = +1 \\ \mathcal{N}(-\mu, \sigma^2 I), & \text{if } y = -1 \end{cases}, \tag{4.1}$$

where the two classes' centers ($\pm\mu \in \mathbb{R}^d$) with each dimension have mean value $\pm\eta$ ($\eta > 0$) and variance σ^2. Formally, we define the data ***separability*** as $S = \eta/\sigma^2$. Intuitively, when S is larger, it suggests that two classes are well separated. Previous work [9] also closely studied this term to describe data separability.

Besides, we assume the imbalanced training dataset satisfying the condition $\Pr.(y = +1) = K \cdot \Pr.(y = -1)$ and $K > 1$, which indicates the imbalance ratio between two classes. During test, we assume two classes have the equal probability to appear. Under the data distribution $\mathcal{D}$, we will discuss the performance of linear classifiers $f(x) = \text{sign}(w^T x - b)$ where w and b are the weight and bias terms of the model f. If a reweighting strategy is involved, we define the model upweights the under-represented class "-1" by ρ.

In the following lemma, we first derive the solution of the optimized linear classifier f training on this imbalanced dataset. Then we will extend the result of Lemma 4.3.1 to analyze the impact of data separability on the performance of model f.

Lemma 4.3.1. *Under the data distribution $\mathcal{D}$ as defined in Eq. (4.1), with an imbalanced ratio K and a reweight ratio ρ, the optimal classifier which minimizes the (reweighted) empirical risk:*

$$\begin{aligned} f^* = \arg\min_f \big(& Pr.(f(x) \neq y | y = -1) \cdot Pr.(y = -1) \cdot \rho \\ & + Pr.(f(x) \neq y | y = +1) \cdot Pr.(y = +1) \big) \end{aligned} \tag{4.2}$$

has the solution: $w = \mathbf{1}$ and $b = \frac{1}{2}\log(\frac{\rho}{K})\frac{d\sigma^2}{\eta} = \frac{1}{2}\log(\frac{\rho}{K})\frac{d}{S}$.

Proof. We will first prove that the optimal model f^* has parameters $w_1 = w_2 = \cdots = w_d$ (or $w = \mathbf{1}$) by contradiction. We define $G = \{1, 2, \ldots, d\}$ and make the following assumption: for the optimal w and b, we assume if there exist $w_i < w_j$ for $i \neq j$ and $i, j \in G$. Then we obtain the following

standard errors for the class “-1" and the class “+1" of this classifier f with weight w:

$$\begin{aligned}
&\Pr.(f^*(x) \neq y | y = -1) = \Pr.(w^T \mathcal{N}(-\eta, \sigma^2) - b > 0) \\
=&\Pr.\{\sum_{k\neq i, k\neq j} w_k \mathcal{N}(-\eta, \sigma^2) + w_i \mathcal{N}(-\eta, \sigma^2) + w_j \mathcal{N}(-\eta, \sigma^2) - b > 0\}, \\
&\Pr.(f^*(x) \neq y | y = +1) = \Pr.(w^T \mathcal{N}(+\eta, \sigma^2) - b < 0) \\
=&\Pr.\{\sum_{k\neq i, k\neq j} w_k \mathcal{N}(+\eta, \sigma^2) + w_i \mathcal{N}(+\eta, \sigma^2) + w_j \mathcal{N}(+\eta, \sigma^2) - b < 0\}.
\end{aligned} \tag{4.3}$$

However, if we define a new classier $\tilde{f}$ whose weight $\tilde{w}$ uses w_j to replace w_i, we obtain the errors for the new classifier:

$$\begin{aligned}
&\Pr.(\tilde{f}(x) \neq y | y = -1) \\
=&\Pr.\{\sum_{k\neq i, k\neq j} w_k \mathcal{N}(-\eta, \sigma^2) + w_j \mathcal{N}(-\eta, \sigma^2) + w_j \mathcal{N}(-\eta, \sigma^2) - b > 0\}, \\
&\Pr.(\tilde{f}(x) \neq y | y = +1) \\
=&\Pr.\{\sum_{k\neq i, k\neq j} w_k \mathcal{N}(+\eta, \sigma^2) + w_j \mathcal{N}(+\eta, \sigma^2) + w_j \mathcal{N}(+\eta, \sigma^2) - b < 0\}.
\end{aligned} \tag{4.4}$$

Comparing the errors in Eq. (4.3) and Eq. (4.4), as $w_i < w_j$, then the classifier $\tilde{f}$ has smaller standard error in each class. Therefore, it contradicts with the assumption that f is the optimal classifier with smallest error. Thus, we conclude for an optimal linear classifier in natural training, it must satisfies $w_1 = w_2 = \cdots = w_d$ (or $w = \mathbf{1}$) if we do not consider the scale of w.

Next, we calculate the optimal bias term b given $w = \mathbf{1}$, where we find an optimal b can minimize the (reweighted) empirical risk:

$$\begin{aligned}
&\text{Error}_{\text{train}}(f^*) \\
=&\Pr.(f^*(x) \neq y | y = -1) \cdot \Pr.(y = -1) \cdot \rho + \Pr.(f^*(x) \neq y | y = +1) \cdot \Pr.(y = +1) \\
\propto&\Pr.(f^*(x) \neq y | y = -1) \cdot \rho + \Pr.(f^*(x) \neq y | y = +1) \cdot K \\
=&\rho \cdot \Pr.(\sum_{i=1}^{d} \mathcal{N}(-\eta, \sigma^2) - b > 0) + K \cdot \Pr.(\sum_{i=1}^{d} \mathcal{N}(\eta, \sigma^2) - b < 0) \\
=&\rho \cdot \Pr.(\mathcal{N}(0,1) < -\frac{b + d\eta}{d\sigma}) + K \cdot \Pr.(\mathcal{N}(0,1) < \frac{b - d\eta}{d\sigma}),
\end{aligned}$$

and we take the derivative with respect to b:

$$\frac{\partial \text{Error}_{\text{train}}}{\partial b} = \frac{\rho}{\sqrt{2\pi}} \cdot (-\frac{1}{d\sigma}) \exp(-\frac{1}{2}(-\frac{b+d\eta}{d\sigma})^2) + \frac{K}{\sqrt{2\pi}} \cdot (\frac{1}{d\sigma}) \exp(-\frac{1}{2}(\frac{b-d\eta}{d\sigma})^2).$$

When $\partial \text{Error}_{\text{train}}/\partial b = 0$, we can calculate the optimal b which gives the minimum value of the empirical error, and we have:

$$b = \frac{1}{2}\log(\frac{\rho}{K})\frac{d\sigma^2}{\eta} = \frac{1}{2}\log(\frac{\rho}{K})\frac{d}{S}.$$

□

Lemma 4.3.1 indicates that the final optimized classifier has a weight vector equal to **1** and its bias term b only depends on K, ρ and the data separability S. In the following, we first focus on one special setting when $\rho = 1$, which is the original ERM model without reweighting. Specifically, we aim to compare the behavior of linear models when they can poorly separate data (like adversarial trained models) or they can well separate data (like naturally trained models).

Theorem 4.3.2. *Under two data distributions $(x^{(1)}, y^{(1)}) \in \mathcal{D}_1$ and $(x^{(2)}, y^{(2)}) \in \mathcal{D}_2$ with different separabilities $S_1 > S_2$, let f_1^* and f_2^* be the optimal non-reweighted classifiers ($\rho = 1$) under $\mathcal{D}_1$ and $\mathcal{D}_2$, respectively. Given the imbalance ratio K is large enough, we have:*

$$\begin{aligned} &Pr.(f_1^*(x^{(1)}) \neq y^{(1)} | y^{(1)} = -1) - Pr.(f_1^*(x^{(1)}) \neq y^{(1)} | y^{(1)} = +1) \\ <\ &Pr.(f_2^*(x^{(2)}) \neq y^{(2)} | y^{(2)} = -1) - Pr.(f_2^*(x^{(2)}) \neq y^{(2)} | y^{(2)} = +1). \end{aligned} \tag{4.5}$$

Proof. Without loss of generality, for distribution $\mathcal{D}_1$, $\mathcal{D}_2$ with different mean-variance pairs $(\pm\eta_1, \sigma_1^2)$ and $(\pm\eta_2, \sigma_2^2)$, we can only consider the case $\eta_1 = \eta_2$ and $\sigma_1^2 < \sigma_2^2$. Otherwise, we can simply rescale one of them to match the mean vector of the other and will not impact the results. Under this definition, the optimal classifier f_1^* and f_2^* has weight vector $w_1 = w_2 = \mathbf{1}$ and bias term b_1, b_2, with the value as demonstrated in Lemma 4.3.1. Next, we will prove the Theorem 4.3.2 by 2 steps.

Step 1. For the error of class "-1", we have:

$$\begin{aligned}
&\Pr.(f_1^*(x^{(1)}) \neq y^{(1)} | y^{(1)} = -1) = \Pr.(\sum_{i=1}^{d} \mathcal{N}(-\eta, \sigma_1^2) - b_1 > 0) \\
&< \Pr.(\sum_{i=1}^{d} \mathcal{N}(-\eta, \sigma_1^2) - b_2 > 0) < \Pr.(\sum_{i=1}^{d} \mathcal{N}(-\eta, \sigma_2^2) - b_2 > 0) \\
&= \Pr.(f_2^*(x^{(2)}) \neq y^{(2)} | y^{(2)} = -1).
\end{aligned}$$

Step 2. For the error of class "+1", we have:

$$\begin{aligned}
&\Pr.(f_1^*(x^{(1)}) \neq y^{(1)} | y^{(1)} = +1) = \Pr.(\sum_{i=1}^{d} \mathcal{N}(\eta, \sigma_1^2) - b_1 < 0) \\
&= \Pr.(\mathcal{N}(0,1) < \frac{b_1 - d\eta}{d\sigma_1}) = \Pr.(\mathcal{N}(0,1) < \frac{-\log(K) \cdot \sigma_1}{2\eta} - \frac{\eta}{\sigma_1}),
\end{aligned} \tag{4.6}$$

and similarly,

$$\Pr.(f_2^*(x^{(2)}) \neq y^{(2)} | y^{(2)} = +1) = \Pr.(\mathcal{N}(0,1) < \frac{-\log(K) \cdot \sigma_2}{2\eta} - \frac{\eta}{\sigma_2}). \tag{4.7}$$

Note that when K is large enough, i.e., $\log(K) > \frac{2 \cdot \eta^2}{\sigma_1 \cdot \sigma_2}$, we can get the Z-score in Eq. (4.6) is larger than Eq. (4.7). As a result, we have:

$$\Pr.(f_1^*(x^{(1)}) \neq y^{(1)} | y^{(1)} = +1) > \Pr.(f_2^*(x^{(2)}) \neq y^{(2)} | y^{(2)} = +1). \tag{4.8}$$

By combining *Step 1* and *Step 2*, we can get the inequality in Theorem 4.3.2. □

Intuitively, Theorem 4.3.2 suggests that when the data separability S is low (such as $\mathcal{D}_2$), the optimized classifier (without reweighting) can intrinsically have a larger error difference between the under-represented class "-1" and the well-represented class "+1". Similar to the observation in Section 4.2.1 and Figure 4.3, adversarially trained models present a weak ability to separate data, and they also present a strong performance gap between the well-represented class and under-represented class. Conclusively, Theorem 4.3.2 indicates that the poor ability to separate the training data can be one important reason which leads to the strong performance gap of adversarially trained models.

Next, we consider the case when the reweighting strategy is applied. Similar to Theorem 4.3.2, we also calculate the models' classwise error under $\mathcal{D}_1$ and $\mathcal{D}_2$ with different levels of separability.

In particular, Theorem 4.3.3 focuses on the well-represented class "+1" and calculates its error increase when upweighting the under-represented class "-1" by ρ. Through the analysis in Theorem 4.3.3, we compare the impact of upweighting the under-represented class on the performance of well-represented class.

Theorem 4.3.3. *Under two data distributions $(x^{(1)}, y^{(1)}) \in \mathcal{D}_1$ and $(x^{(2)}, y^{(2)}) \in \mathcal{D}_2$ with different separabilities $S_1 > S_2$, let f_1^* and f_2^* be the optimal non-reweighted classifiers ($\rho = 1$) under $\mathcal{D}_1$ and $\mathcal{D}_2$, respectively, and let $f_1'^*$ and $f_2'^*$ be the optimal reweighted classifiers under $\mathcal{D}_1$ and $\mathcal{D}_2$ given the optimal reweighting ratio ($\rho = K$). Given the imbalance ratio K is large enough, we have:*

$$\begin{aligned} &Pr.(f_1'^*(x^{(1)}) \neq y^{(1)} | y^{(1)} = +1) - Pr.(f_1^*(x^{(1)}) \neq y^{(1)} | y^{(1)} = +1) \\ &< Pr.(f_2'^*(x^{(2)}) \neq y^{(2)} | y^{(2)} = +1) - Pr.(f_2^*(x^{(2)}) \neq y^{(2)} | y^{(2)} = +1). \end{aligned} \tag{4.9}$$

Proof. We first show that under both distribution $\mathcal{D}_1$ and $\mathcal{D}_2$, the optimal reweighting ratio ρ is equal to the imbalance ratio K. Based on the results in Eq. (4.3) and calculated model parameters w and b, we have the test error (given the model trained by reweight value ρ):

$$\begin{aligned} &\text{Error}_{\text{test}}(f^*) \\ &= \text{Pr.}(f^*(x) \neq y | y = -1) \cdot \text{Pr.}(y = -1) + \text{Pr.}(f^*(x) \neq y | y = +1) \cdot \text{Pr.}(y = +1) \\ &\propto \text{Pr.}(\mathcal{N}(0,1) < -\frac{b + d\eta}{d\sigma}) + \text{Pr.}(\mathcal{N}(0,1) < \frac{b - d\eta}{d\sigma}) \\ &= \text{Pr.}(\mathcal{N}(0,1) < -\frac{1}{2}\log(\frac{\rho}{K}) - \frac{\sigma}{\eta}) + \text{Pr.}(\mathcal{N}(0,1) < \frac{1}{2}\log(\frac{\rho}{K}) - \frac{\sigma}{\eta}). \end{aligned}$$

The value of taking the minimum when its derivative with respect to ρ is equal to 0, where we can get $\rho = K$ and the bias term $b = 0$. Note that the variance values have the relation: $\sigma_1^2 < \sigma_2^2$. Therefore, it is easy to get that:

$$\begin{aligned} &\text{Pr.}(f_1'^*(x^{(1)}) \neq y^{(1)} | y^{(1)} = +1) = \text{Pr.}(\sum_{i=1}^{d} \mathcal{N}(\eta, \sigma_1^2) < 0) \\ &< \text{Pr.}(\sum_{i=1}^{d} \mathcal{N}(\eta, \sigma_2^2) < 0) = \text{Pr.}(f_2'^*(x^{(2)}) \neq y^{(2)} | y^{(2)} = +1). \end{aligned} \tag{4.10}$$

Combining the results in Eq. (4.8) and (4.10), we have proved the inequality in Theorem 4.3.3. □

As Theorem 4.3.3 shows, when the data distribution has poorer data separability (such as $\mathcal{D}_2$), upweighting the under-represented class can cause greater hurt on the performance of the well-represented class. It is also consistent with our empirical findings about adversarial training models. Since the adversarially trained models poorly separate the data (Figure 4.3), upweighting the under-represented class always drastically decreases the performance of the well-represented class (Section 4.2.2). Through the discussions in both Theorem 4.3.2 and Theorem 4.3.3, we conclude that the poor separability can be one important reason which makes adversarial training and its reweighted variants extremely difficult to achieve good performance under imbalance data distribution. Therefore, in the next section, we will explore potential solutions which can facilitate the reweighting strategy in adversarial training.

4.4 Separable Reweighted Adversarial Training

The observations from both preliminary study and theoretical understandings indicate that more separable data will advance the reweighting strategy in adversarial training under imbalanced scenarios. Thus, in this section, we present a framework, Separable Reweighted Adversarial Training (SRAT), which enables the effectiveness of the reweighting strategy in adversarial training under imbalanced scenarios by increasing the separability in the learned feature space.

4.4.1 Reweighted Adversarial Training

Given an input example (x, y), adversarial training [71] aims to obtain a robust model f_θ that can make the same prediction y for an adversarial example x', generated by applying an adversarially perturbation on x. The adversarial perturbations are typically bounded by a small value ϵ under L_p-norm, i.e., $\|x' - x\|_p \leq \epsilon$.

As indicated in Section 4.2.1, adversarial training cannot be applied in imbalanced scenarios directly, as it presents very low performance on under-represented classes. To tackle this problem, a natural idea is to integrate existing imbalanced learning strategies proposed in natural training, such as reweighting, into adversarial training to improve the trained model's performance on those

under-represented classes. Hence, the reweighted adversarial training can be defined as

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \max_{\|x_i' - x_i\|_p \leq \epsilon} w_i \mathcal{L}(f_\theta(x_i'), y_i), \tag{4.11}$$

where w_i is a weight value assigned for each input sample (x_i, y_i) based on the example size of the class (x_i, y_i) belongs to or some properties of (x_i, y_i). In most existing adversarial training methods [71, 111, 103], the cross entropy (CE) loss is adopted as the loss function $\mathcal{L}(\cdot, \cdot)$. However, the CE loss could be suboptimal in imbalanced scenarios and some new loss functions designed for imbalanced learning specifically, such as Focal loss [63] and LDAM loss [10], have been proven their superiority in natural training. Hence, besides CE loss, Focal loss and LDAM loss can also be adopted as the loss function $\mathcal{L}(\cdot, \cdot)$ in Eq. (4.11).

4.4.2 Increasing Feature Separability

Our preliminary study indicates that only reweighted adversarial training cannot work well under imbalanced scenarios. Moreover, the reweighting strategy behaves very differently between natural training and adversarial training. Meanwhile, our theoretical analysis suggests that the poor separability of the feature space produced by the adversarially trained model can be one reason to understand these observations. Hence, in order to facilitate the reweighting strategy in adversarial training under imbalanced scenarios, we equip a feature separation loss with our SRAT method. We aim to enforce the learned feature space as separable as possible. More specifically, the goal of the feature separation loss is to make (1) the learned features of examples from the same class well clustered, and (2) the features of examples from different classes well separated. By achieving this goal, the model is able to learn more discriminative features for each class. Correspondingly adjusting the decision boundary via the reweighting strategy to fit under-represented classes' examples more will not hurt well-represented classes drastically. The feature separation loss is formally defined as:

$$\mathcal{L}_{sep}(x_i') = -\frac{1}{|P(i)|} \sum_{p \in P(i)} \log \frac{\exp(z_i' \cdot z_p'/\tau)}{\sum_{a \in A(i)} \exp(z_i' \cdot z_a'/\tau)}, \tag{4.12}$$

where $\mathbf{z}_i'$ is the feature representation of the adversarial example $\mathbf{x}_i'$ of $\mathbf{x}_i$, $\tau \in \mathcal{R}^+$ is a scalar temperature parameter, $P(i)$ denotes the set of input examples belonging to the same class with $\mathbf{x}_i$

and $A(i)$ indicates the set of all input examples excepts $\mathbf{x}'_i$. When minimizing the feature separation loss during training, the learned features of examples from the same class will tend to aggregate together in the latent feature space, and, hence, result in a more separable latent feature space. Our proposed feature separation loss $\mathcal{L}_{sep}(\cdot)$ is inspired by the supervised contrastive loss proposed in [52]. The main difference is, instead of applying data augmentation techniques to generate two different views of each data example and feeding the model with augmented data examples, our feature separation loss directly takes the adversarial example $\mathbf{x}'_i$ of each data example $\mathbf{x}_i$ as input.

4.4.3 Training Schedule

By combining the feature separation loss with the reweighted adversarial training, the final object function for Separable Reweighted Adversarial Training (SRAT) is defined as:

$$\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \max_{\|x'_i - x_i\|_p \leq \epsilon} w_i \mathcal{L}(f_\theta(x'_i), y_i) + \lambda \mathcal{L}_{sep}(x'_i), \tag{4.13}$$

where we use a hyper-parameter λ to balance the contributions from the reweighted adversarial training and the feature separation loss.

In practice, in order to better take advantage of the reweighting strategy in our SRAT method, we adopt a deferred reweighting training schedule [10]. Specifically, before annealing the learning rate, our SRAT method first trains a model guided by Eq. (4.13) without introducing the reweighting strategy, i.e., setting $w_i = 1$ for every input example $\mathbf{x}'_i$, and then applies reweighting into model training process with a smaller learning rate. Our SRAT method enables to learn more separable feature space, thus comparing with applying the reweighting strategy from the beginning of training, this deferred re-balancing training schedule enables the reweighting strategy to obtain more benefits from our SRAT method, and as a result, it can boost the performance of our SRAT method with the help of the reweighting strategy.

4.4.4 Algorithm

The algorithm of our proposed SRAT framework is shown in Algorithm 4.1. Specifically, in each training iteration, we first generate adversarial examples using PGD for examples in the current batch (Line 5). If the current training iteration does not reach a predefined starting reweighting

Algorithm 4.1 Separable Reweighted Adversarial Training.

Input: imbalanced training dataset $D = \{(x_i, y_i)\}_{i=1}^{n}$, number of total training epochs T, starting reweighting epoch T_d, batch size N, number of batches M, learning rate γ

Output: adversarially robust model f_θ

1: Initialize the model parameters θ randomly.
2: **for** epoch = $1, \ldots, T_d - 1$ **do**
3: **for** mini-batch = $1, \ldots, M$ **do**
4: Sample a mini-batch $\mathcal{B} = \{(x_i, y_i)\}_{i=1}^{N}$ from D.
5: Generate adversarial example x_i' for each $x_i' \in \mathcal{B}$.
6: $\mathcal{L}(f_\theta) = \frac{1}{N}\sum_{i=1}^{N} \max_{\|x_i'-x_i\|_p \le \epsilon} \mathcal{L}(f_\theta(x_i'), y_i) + \lambda \mathcal{L}_{sep}(x_i')$
7: $\theta \leftarrow \theta - \gamma \nabla_\theta \mathcal{L}(f_\theta)$
8: **end for**
9: Optional: $\gamma \leftarrow \gamma / \kappa$
10: **end for**
11: **for** epoch = $T_d, \ldots, T$ **do**
12: **for** mini-batch = $1, \ldots, M$ **do**
13: Sample a mini-batch $\mathcal{B} = \{(x_i, y_i)\}_{i=1}^{N}$ from D.
14: Generate adversarial example x_i' for each $x_i' \in \mathcal{B}$.
15: $\mathcal{L}(f_\theta) = \frac{1}{N}\sum_{i=1}^{N} \max_{\|x_i'-x_i\|_p \le \epsilon} w_i \mathcal{L}(f_\theta(x_i'), y_i) + \lambda \mathcal{L}_{sep}(x_i')$
16: $\theta \leftarrow \theta - \gamma \nabla_\theta \mathcal{L}(f_\theta)$
17: **end for**
18: Optional: $\gamma \leftarrow \gamma / \kappa$
19: **end for**

epoch T_d, we will assign same weights, i.e., $w_i = 1$ for all adversarial examples x_i in the current batch (Line 6). Otherwise, the reweighting strategy will be adopted in the final loss function (Line 15), where a specific weight w_i will be assigned for each adversarial example x_i if its corresponding clean example x_i comes from an under-represented class.

4.5 Experiment

In this section, we perform experiments to validate the effectiveness of our SRAT method. We first compare SRAT with several representative imbalanced learning methods in adversarial training under various imbalanced scenarios and then conduct ablation study to deeply understand SRAT.

4.5.1 Experimental Settings

Data sets. We conduct experiments on multiple imbalanced training datasets artificially created from two benchmark image datasets CIFAR10, and CIFAR100 [56] with diverse imbalanced

distributions. Specifically, we consider two different imbalance types: Exponential (Exp) imbalance [21] and Step imbalance [7]. For Exp imbalance, the number of training examples of each class will be reduced according to an exponential function $n = n_i\tau^i$, where i is the class index, n_i is the number of training examples in the original training dataset for class i and $\tau \in (0, 1)$. We categorize half classes with most frequent example sizes in the imbalanced training dataset as well-represented classes and the remaining half classes as under-represented classes. For Step imbalance, we follow the similar process adopted in Section 4.2.1. Moreover, we denote *imbalance ratio* K as the ratio between training example sizes of the most frequent and least frequent class. We construct different imbalanced datasets "Step-10", "Step-100", "Exp-10" and "Exp-100", by adopting different imbalanced types (Step or Exp) with different imbalanced ratios ($K = 10$ or $K = 100$) to train models, and evaluate model's performance on the original uniformly distributed test datasets of CIFAR10 and CIFAR100 correspondingly.

Baseline methods. We implement several representative and state-of-the-art imbalanced learning methods (or their combinations) into adversarial training as baseline methods. These methods include: (1) Focal loss (Focal); (2) LDAM loss (LDAM); (3) Class-balanced reweighting (CB-Reweight) [21], where each example is reweighted proportionally by the inverse of the effective number[2] of its class; (4) Class-balanced Focal loss (CB-Focal) [21], a combination of Class-balanced method and Focal loss, where well-classified examples will be downweighted while hard-classified examples will be upweighted controlled by their corresponding effective number; (5) deferred reweighted CE loss (DRCB-CE), where a deferred reweighting training schedule is applied based on the CE loss; (6) deferred reweighted Class-balanced Focal loss (DRCB-Focal), where a deferred reweighting training schedule is applied based on the CB-Focal loss; (7) deferred reweighted Class-balanced LDAM loss (DRCB-LDAM) [10], where a deferred reweighting training schedule is applied based on the CB-LDAM loss. We also include the original PGD adversarial training method using cross entropy loss (CE) in our experiments.

Our proposed methods. We evaluate three variants of our proposed SRAT method with

[2]The effective number is defined as the volume of examples and can be calculated by $(1 - \beta^{n_i})/(1 - \beta)$, where $\beta \in [0, 1)$ is a hyperparameter and n_i denotes the number of examples of class i.

Table 4.1 Performance comparison on the CIFAR10 Step-10 dataset under l_∞ threat model.

Metric	Standard Accuracy		Robust Accuracy	
Method	Overall	Under	Overall	Under
CE	63.26 ± 0.59	40.62 ± 1.10	36.96 ± 0.36	14.23 ± 0.83
Focal	63.57 ± 0.92	41.17 ± 2.07	36.89 ± 0.36	14.25 ± 0.97
LDAM	57.08 ± 1.16	31.09 ± 2.20	37.18 ± 0.56	12.44 ± 0.93
CB-Reweight	73.30 ± 0.30	74.80 ± 0.88	41.34 ± 0.42	42.15 ± 1.42
CB-Focal	73.42 ± 0.29	74.35 ± 1.39	41.34 ± 0.23	41.80 ± 1.24
DRCB-CE	75.89 ± 0.23	70.55 ± 1.10	39.93 ± 0.24	33.33 ± 1.42
DRCB-Focal	74.61 ± 0.35	67.06 ± 1.37	37.91 ± 0.24	29.50 ± 1.31
DRCB-LDAM	72.95 ± 0.08	75.42 ± 1.83	45.23 ± 0.19	44.98 ± 1.90
SRAT-CE	**76.69 ± 0.33**	73.07 ± 0.63	41.02 ± 0.49	36.57 ± 0.92
SRAT-Focal	75.41 ± 0.69	74.91 ± 0.70	42.05 ± 0.52	41.28 ± 0.82
SRAT-LDAM	73.99 ± 0.52	**76.63 ± 0.39**	**45.60 ± 0.18**	**45.95 ± 0.51**

Table 4.2 Performance comparison on the CIFAR10 Step-100 dataset under l_∞ threat model.

Metric	Standard Accuracy		Robust Accuracy	
Method	Overall	Under	Overall	Under
CE	47.29 ± 0.32	9.03 ± 0.99	30.39 ± 0.24	1.62 ± 0.41
Focal	47.36 ± 0.19	9.03 ± 0.52	30.12 ± 0.31	1.45 ± 0.12
LDAM	42.49 ± 0.62	0.85 ± 0.46	30.80 ± 0.31	0.05 ± 0.06
CB-Reweight	37.68 ± 1.18	19.64 ± 1.82	25.58 ± 0.62	10.33 ± 0.82
CB-Focal	15.44 ± 3.85	0.00 ± 0.00	14.46 ± 3.16	0.00 ± 0.00
DRCB-CE	53.40 ± 1.20	22.86 ± 3.03	28.31 ± 0.59	3.35 ± 0.56
DRCB-Focal	52.75 ± 0.96	21.81 ± 2.27	27.78 ± 0.49	3.24 ± 0.57
DRCB-LDAM	61.60 ± 0.44	50.69 ± 2.27	31.37 ± 0.45	16.25 ± 2.04
SRAT-CE	60.04 ± 1.16	41.71 ± 2.07	30.00 ± 0.80	12.25 ± 1.43
SRAT-Focal	62.93 ± 1.10	51.83 ± 3.33	28.38 ± 1.00	15.89 ± 3.15
SRAT-LDAM	**63.13 ± 1.17**	**52.73 ± 3.23**	**33.51 ± 0.68**	**18.89 ± 0.59**

different implementations of the prediction loss $\mathcal{L}(\cdot,\cdot)$ in Eq. (4.11), i.e., CE loss, Focal loss and LDAM loss. The variant utilizing CE loss is denoted as SRAT-CE, and, similarly, other two variants are denoted as SRAT-Focal and SRAT-LDAM, respectively. For all these three variants, Class-balanced method [21] is adopted to set weight values within the deferred reweighting training schedule.

Implementation details. All aforementioned methods are implemented using a Pytorch library DeepRobust [62]. For CIFAR10/CIFAR100 based datasets, the adversarial examples used in training are calculated by PGD-10, with a perturbation budget $\epsilon = 8/255$ and step size $\gamma = 2/255$;

Table 4.3 Performance comparison on the CIFAR10 Exp-10 dataset under l_∞ threat model.

Metric	Standard Accuracy		Robust Accuracy	
Method	Overall	Under	Overall	Under
CE	71.95 ± 0.52	64.09 ± 0.44	37.94 ± 0.19	26.79 ± 0.51
Focal	72.06 ± 0.78	63.99 ± 1.15	37.62 ± 0.34	26.27 ± 1.04
LDAM	67.39 ± 1.00	58.01 ± 2.26	41.35 ± 0.32	28.65 ± 0.83
CB-Reweight	75.17 ± 0.15	76.87 ± 0.69	41.02 ± 0.39	41.67 ± 0.89
CB-Focal	74.73 ± 0.41	76.67 ± 0.26	38.86 ± 0.67	42.41 ± 0.56
DRCB-CE	76.25 ± 0.09	75.83 ± 0.49	40.02 ± 0.45	37.93 ± 0.65
DRCB-Focal	75.36 ± 0.40	72.72 ± 0.94	37.76 ± 0.54	33.83 ± 0.68
DRCB-LDAM	73.92 ± 0.31	78.53 ± 1.24	46.29 ± 0.46	48.81 ± 0.54
SRAT-CE	**76.74 ± 0.15**	78.61 ± 0.63	42.39 ± 0.71	43.37 ± 0.38
SRAT-Focal	75.26 ± 0.00	**80.52 ± 0.00**	42.37 ± 0.00	47.22 ± 0.00
SRAT-LDAM	74.63 ± 0.00	79.82 ± 0.00	**46.72 ± 0.00**	**50.38 ± 0.00**

Table 4.4 Performance comparison on the CIFAR10 Exp-100 dataset under l_∞ threat model.

Metric	Standard Accuracy		Robust Accuracy	
Method	Overall	Under	Overall	Under
CE	48.40 ± 0.59	23.04 ± 1.15	26.94 ± 0.84	6.17 ± 0.86
Focal	49.16 ± 0.61	23.69 ± 1.15	26.84 ± 0.59	5.88 ± 0.48
LDAM	48.39 ± 0.99	25.69 ± 1.35	29.51 ± 0.27	8.95 ± 0.45
CB-Reweight	57.49 ± 0.58	56.47 ± 1.67	29.01 ± 0.30	26.53 ± 1.27
CB-Focal	50.35 ± 0.44	60.05 ± 0.53	27.15 ± 0.20	**33.56 ± 0.35**
DRCB-CE	57.30 ± 0.30	37.90 ± 1.23	26.97 ± 0.55	10.57 ± 1.03
DRCB-Focal	54.76 ± 0.30	31.79 ± 1.30	25.24 ± 0.39	7.81 ± 0.87
DRCB-LDAM	62.65 ± 0.50	57.19 ± 2.10	31.66 ± 0.56	22.11 ± 1.70
SRAT-CE	**64.29 ± 0.46**	61.81 ± 1.83	29.99 ± 0.43	24.09 ± 0.98
SRAT-Focal	62.57 ± 0.47	64.88 ± 0.81	30.34 ± 0.67	28.66 ± 1.60
SRAT-LDAM	63.11 ± 0.08	**65.60 ± 1.94**	**34.22 ± 0.41**	32.55 ± 1.70

in evaluation, we report robust accuracy under l_∞-norm $8/255$ attacks generated by PGD-20 on Resnet-18 [42] models. We set the total training epochs to 200 and the initial learning rate to 0.1, and decay the learning rate at epoch 160 and 180 with the ratio 0.01. The deferred reweighting strategy will be applied starting from epoch 160.

4.5.2 Performance Comparison

Table 4.1 and Table 4.6 show the performance comparison on several different imbalanced CIFAR10 and CIFAR100 data sets. In these two tables, we use bold values to denote the highest accuracy among all methods and use the underline values to indicate our SRAT variants which

Table 4.5 Performance comparison on the CIFAR100 Step-10 dataset under l_∞ threat model.

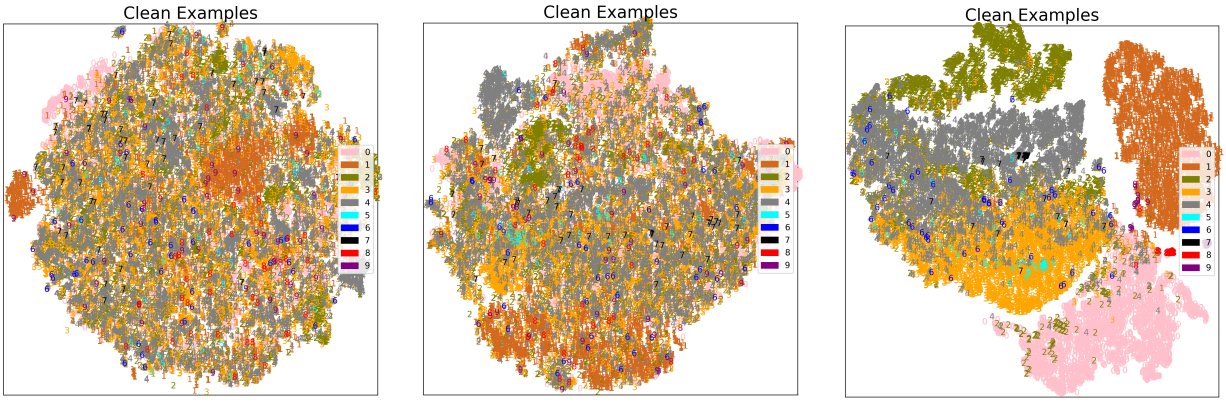
Metric	Standard Accuracy		Robust Accuracy	
Method	Overall	Under	Overall	Under
CE	39.90 ± 0.11	17.90 ± 0.38	17.88 ± 0.32	6.40 ± 0.60
Focal	40.10 ± 0.27	17.99 ± 0.75	17.67 ± 0.30	6.40 ± 0.18
LDAM	39.34 ± 0.54	17.57 ± 0.94	20.95 ± 0.20	7.41 ± 0.37
CB-Reweight	38.88 ± 0.40	30.73 ± 0.49	16.67 ± 0.58	11.71 ± 0.62
CB-Focal	39.49 ± 0.30	28.96 ± 0.14	16.55 ± 0.39	11.09 ± 0.33
DRCB-CE	45.21 ± 0.11	33.26 ± 0.09	18.36 ± 0.33	11.15 ± 0.48
DRCB-Focal	44.28 ± 0.15	30.57 ± 0.22	17.30 ± 0.39	9.73 ± 0.18
DRCB-LDAM	44.70 ± 0.46	35.90 ± 0.92	21.80 ± 0.12	15.19 ± 0.36
SRAT-CE	**47.17 ± 0.26**	37.81 ± 0.38	21.36 ± 0.31	15.41 ± 0.19
SRAT-Focal	46.83 ± 0.28	**38.10 ± 0.58**	21.66 ± 0.32	16.52 ± 0.32
SRAT-LDAM	45.41 ± 0.55	36.39 ± 0.65	**23.15 ± 0.15**	**16.84 ± 0.08**

Table 4.6 Performance comparison on the CIFAR100 Exp-10 dataset under l_∞ threat model.

Metric	Standard Accuracy		Robust Accuracy	
Method	Overall	Under	Overall	Under
CE	41.88 ± 0.36	31.30 ± 0.57	16.62 ± 0.03	11.22 ± 0.21
Focal	41.64 ± 0.51	31.02 ± 0.71	16.29 ± 0.18	10.97 ± 0.34
LDAM	41.55 ± 0.60	31.74 ± 0.91	20.20 ± 0.20	14.71 ± 0.51
CB-Reweight	41.82 ± 0.11	34.37 ± 0.31	17.05 ± 0.35	13.53 ± 0.57
CB-Focal	40.86 ± 0.13	32.21 ± 0.01	16.08 ± 0.41	12.30 ± 0.59
DRCB-CE	43.89 ± 0.26	37.28 ± 0.29	16.90 ± 0.19	13.62 ± 0.14
DRCB-Focal	43.38 ± 0.30	36.17 ± 0.57	16.04 ± 0.18	12.56 ± 0.27
DRCB-LDAM	43.36 ± 0.48	39.27 ± 0.72	20.36 ± 0.30	17.63 ± 0.38
SRAT-CE	45.84 ± 0.18	41.72 ± 0.53	21.20 ± 0.15	**19.23 ± 0.36**
SRAT-Focal	**46.38 ± 0.28**	**42.53 ± 0.79**	20.09 ± 0.25	17.83 ± 0.56
SRAT-LDAM	44.98 ± 0.33	40.39 ± 0.69	**21.83 ± 0.33**	18.99 ± 0.59

achieve the highest accuracy among their corresponding baseline methods utilizing the same loss function for making predictions.

From these tables, we can make the following observations. First, compared to baseline methods, our SRAT method obtains improved performance in terms of both overall standard & robust accuracy under almost all imbalanced settings. More importantly, SRAT makes significant improvements on those under-represented classes, especially under the extremely imbalanced settings. For example, on the CIFAR10 Step-100 data set, our SRAT-Focal method improves the standard accuracy on under-represented classes from 21.81% achieved by the best baseline method utilizing



(a) CE. (b) DRCB-LDAM. (c) SRAT-LDAM.

Figure 4.4 t-SNE visualization of different learned features.

Focal loss to 51.83% and robust accuracy from 3.24% to 15.89%. These results demonstrate that SRAT is able to obtain more robustness under imbalanced settings. Second, the performance gap among three SRAT variants are mainly caused by the gap between the loss functions in these methods. As shown in these two tables, DRCB-LDAM typically performs better than DRCE-CE and DRCB-Focal, and similarly, SRAT-LDAM outperforms SRAT-CE and SRAT-Focal under the same settings.

4.5.3 Ablation Study

In this subsection, we provide ablation study to understand our SRAT method more comprehensively.

Feature space visualization. In order to facilitate the reweighting strategy in adversarial training under the imbalanced setting, we present a feature separation loss in our SRAT method. The main goal of the feature separation loss is to enforce the learned feature space as much separated as possible. For checking whether the feature separation loss can work as expected, we apply t-SNE [94] to visualize the latent feature space learned by our SRAT-LDAM method as well as by original PGD adversarial training method (CE) and DRCB-LDAM method in Figure 4.4.

As shown in Figure 4.4, the feature space learned by our SRAT-LDAM method is more separable than two baseline methods, which demonstrates that, with our feature separation loss, the adversarially trained model is able to learn much better features and thus SRAT can achieve better

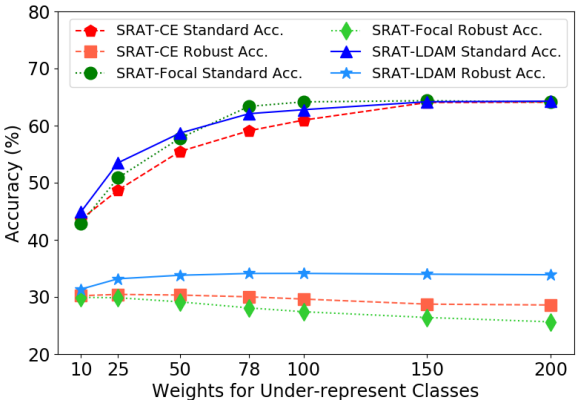


Figure 4.5 The impact of weights.

performance.

Impact of weight values. As in all SRAT variants, we adopt the Class-balanced method [21] to assign different weights to different classes. To explore how the assigned weights impact the performance of SRAT, we conduct experiments using CIFAR10 Step-100 dataset to see the change of model's performance using different reweighting values. Specifically, we assign well-represented classes with weight 1 and change the weight for under-represented classes from 10 to 200. The experimental results are shown in Figure 4.5. Here, we use an approximated value 78 to denote the weight calculated by the Class-balanced method when the imbalance ratio equals 100.

From Figure 4.5, we can obverse that, for all SRAT variants, the model's standard accuracy is increased with the increasing of the weights for under-represented classes. However, the robust accuracy for these three methods do not synchronize with the change of their standard accuracy. When increasing the weights for under-represented classes, robust accuracy of SRAT-LDAM is almost unchanged and of SRAT-CE and SRAT-Focal even has slight decrease. As a trade-off, using a relative large weight, such as 78 or 100, in SRAT can obtain satisfactory performance on both standard & robust accuracy.

Impact of hyper-parameter λ. In our SRAT method, the contributions of feature separation loss and prediction loss are controlled by a hyper-parameter λ. In this part, we study how this hyper-parameter affects the performance of SRAT. In experiments, we evaluate the models' performance of all SRAT variants with different values of λ used in training process on CIFAR10 Step-100

dataset.

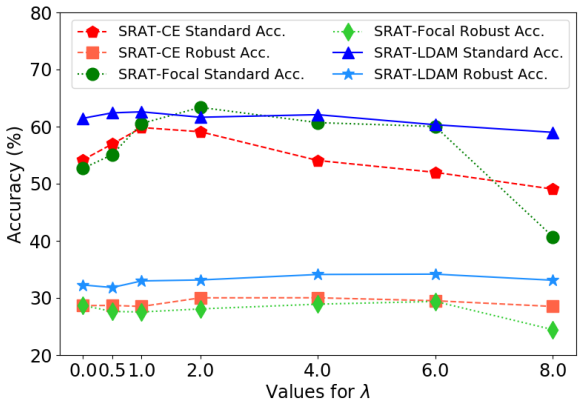


Figure 4.6 The impact of λ.

As shown in Figure 4.6, the performance of all SRAT variants are not very sensitive with the choice of λ. However, a large value of λ, such as 8, may hurt the model's performance.

Impact of imbalance ratio K. In previous experiments, we evaluate the effectiveness of our SRAT method using various imbalanced datasets with imbalance ratio $K = 10$ or $K = 100$. To investigate the performance of our SRAT method more comprehensively, in this part, we test our SRAT method on more imbalanced datasets with diverse imbalance ratios. Specifically, we construct a series of "Step" imbalanced CIFAR10 datasets by setting the value of the imbalance ratio K from 5 to 100. For comparison, we apply both DRCB-Focal method and our SRAT-Focal variant to train models on those imbalanced datasets and test the trained models' performance on the original uniformly distributed CIFAR10 test dataset. The experimental results are shown in

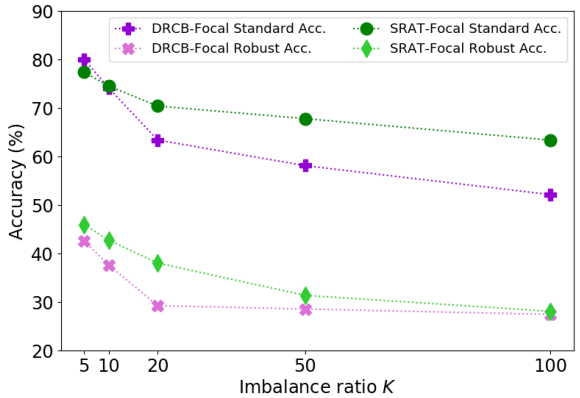


Figure 4.7 The impact of K.

Table 4.7 Performance comparison on the CIFAR10 Step-100 dataset under l_2 threat model.

Metric	Standard Accuracy		Robust Accuracy	
Method	Overall	Under	Overall	Under
CE	65.01 ± 1.84	35.79 ± 3.72	52.22 ± 1.99	20.07 ± 3.48
Focal	66.15 ± 2.75	38.77 ± 5.80	55.02 ± 3.13	24.67 ± 5.99
LDAM	57.35 ± 2.47	20.25 ± 4.37	52.11 ± 2.14	15.49 ± 3.44
CB-Reweight	64.32 ± 0.85	40.75 ± 2.47	52.72 ± 0.71	27.47 ± 1.75
CB-Focal	65.89 ± 0.82	45.65 ± 2.01	55.81 ± 1.31	33.85 ± 2.64
DRCB-CE	70.78 ± 1.84	48.57 ± 4.01	56.00 ± 2.00	30.39 ± 4.01
DRCB-Focal	71.59 ± 1.21	50.85 ± 2.60	57.89 ± 1.88	34.14 ± 3.31
DRCB-LDAM	71.51 ± 1.32	50.99 ± 1.85	64.68 ± 1.15	40.55 ± 1.75
SRAT-CE	**76.27 ± 1.46**	**60.76 ± 3.04**	61.83 ± 1.53	42.72 ± 2.93
SRAT-Focal	73.73 ± 0.48	54.68 ± 1.06	60.12 ± 0.54	38.01 ± 1.35
SRAT-LDAM	73.89 ± 0.78	57.09 ± 2.43	**67.38 ± 0.92**	**47.45 ± 2.75**

Figure 4.7.

From Figure 4.7, we can obverse that, under different imbalanced scenarios, the model trained by our SRAT-Focal method can always achieve better performance than the one trained by DRCB-Focal method. In other words, the effectiveness of our SRAT method will not be affected by the imbalanced ratio K, which determines the data distribution of the imbalanced training dataset.

4.5.4 Performance under l_2 Threat Model

To further evaluate the effectiveness of our SRAT method, we also adversarially train Resnet-18 [42] models on CIFAR10 Step-100 dataset under l_2 attack. We follow the same settings in [105], where the perturbation budge $\epsilon = 128/255$ and step size $\gamma = 15/255$. As shown in Table 4.7, SRAT outperforms all baseline methods with a large margin, which verifies the effectiveness of SRAT.

4.6 Related Work

Adversarial Robustness. The vulnerability of DNN models to adversarial examples has been verified by many existing successful attack methods [32, 11]. To improve model robustness against adversarial attacks, various defense methods have been proposed [71, 78, 19]. Among them, adversarial training has been proven to be one of the most effective defense methods [4]. Adversarial training can be formulated as solving a min-max optimization problem where the

outer minimization process enforces the model to be robust to adversarial examples, generated by the inner maximization process via some existing attacking methods like PGD [71]. Based on adversarial training, several variants, such as TRADES [111], MART [103], have been presented to improve the model's performance further. More details about adversarial robustness can be found in recent surveys [13, 109]. Since almost all studies of adversarial training are focused on balanced datasets, it's worthwhile to investigate the performance of adversarial training methods on imbalanced training datasets.

Imbalanced Learning. Most existing works of imbalanced training can be roughly classified into two categories, i.e., re-sampling and reweighting. *Re-sampling* methods aim to reduce imbalance level through either over-sampling examples from under-represented classes [7, 9] or under-sampling examples from well-represented classes [46, 25, 40]. *Reweighting* methods allocate different weights for different classes or even different examples. For example, Focal loss [63] enlarges the weights of wrongly-classified examples while reducing the weights of well-classified examples in the standard cross entropy loss; and LDAM loss [10] regularizes the under-represented classes more strongly than the well-represented classes to attain good generalization on under-represented classes. More details about imbalanced learning can be found in recent surveys [41, 47]. The majority of existing methods focused on the nature training scenario and their trained models will be crashed when facing adversarial attacks [91, 32]. Hence, in this chapter, we develop a novel method that can defend adversarial attacks and achieve well-pleasing performance under imbalanced scenarios.

4.7 Chapter Conclusion

In this chapter, we first empirically investigate the behavior of adversarial training under imbalanced scenarios and explore potential solutions to assist adversarial training in tackling the imbalanced issue. As neither adversarial training itself nor adversarial training with reweighting can work well under imbalanced scenarios, we further theoretically verify the poor data separability is one key reason causing the failure of adversarial training based methods. Based on our findings, we propose the Separable Reweighted Adversarial Training (SRAT) method to facilitate

the reweighting strategy in imbalanced adversarial training. We validate the effectiveness of SRAT via extensive experiments. In the future, we plan to examine how other types of defense methods perform under imbalanced scenarios and how other types of balanced learning methods behave under adversarial training.

CHAPTER 5

MIX-UP STRATEGY TO ENHANCE ADVERSARIAL TRAINING WITH IMBALANCED DATA

Adversarial training has been proven to be one of the most effective techniques to defend against adversarial examples. The majority of existing adversarial training methods assume that every class in the training data is equally distributed. However, in reality, some classes often have a large number of training data while others only have a very limited amount. Recent studies have shown that the performance of adversarial training will degrade drastically if the training data is imbalanced. In this chapter, we propose a simple yet effective framework to enhance the robustness of DNN models under imbalanced scenarios. Our framework, *Imb-Mix*, first augments the training dataset by generating multiple adversarial examples for samples in the minority classes. This is done by first adding random noise to the original adversarial examples created by one specific adversarial attack method. It then constructs Mixup-mimic mixed examples upon the augmented dataset used by adversarial training. In addition, we theoretically prove the regularization effect of our Mixup-mimic mixed examples generation technique in Imb-Mix. Extensive experiments on various imbalanced datasets verify the effectiveness of the proposed framework.

5.1 Chapter Introduction

Deep neural networks (DNNs) have been successfully applied in a wide range of real-world applications, such as computer vision [42], natural language processing [96] and speech recognition [1]. However, DNNs are highly vulnerable to adversarial examples [32, 11]. By adding an imperceptible amount of noise to benign examples, manually crafted adversarial examples can mislead a well-trained DNN based classifier. This can cause the classifier to incorrectly classify benign samples, with high confidence, that it previously classified correctly. Due to the large threat of adversarial examples, considerable efforts have been made to improve the robustness of DNNs. Among them, adversarial training [71, 111] has been empirically proven to be one of the most effective and reliable defense methods. Generally, adversarial training can be formulated as a min-max optimization problem where the inner maximization process generates adversarial

examples that can mostly fool the model, and the outer minimization process reduces the model's average classification error on the generated adversarial examples.

Although they have been shown to improve the robustness of DNNs, most existing adversarial training methods assume that the number of training examples from each class is balanced. However, this assumption does not hold in many real-world applications where some classes can have a notably larger presence than other classes [95, 69]. Hence, the training data is typically imbalanced among classes. Very recently, there have been works [106, 102] that examine adversarial training under imbalanced scenarios. They've shown that in such situations, adversarial training will lead to a huge performance discrepancy between classes with more training examples (i.e., *majority classes*) and classes with fewer training examples (i.e., *minority classes*). Furthermore, it cannot provide satisfactory robustness for those minority classes. Therefore, it is natural to ask: *How can we improve adversarial training under imbalanced scenarios?* Since imbalanced training data often causes the trained classifier to be overwhelmed by the majority classes and ignore the minority classes, two common ways to alleviate the negative impacts are re-sampling and re-weighting. Re-sampling attempts to balance the data distribution [15, 26] by upsampling minority class samples or downsampling majority class samples. Re-weighting, assigns higher weights in the loss to samples from the minority class to make the trained model to be biased toward minority classes [63, 21, 10]. The majority of existing works only consider the imbalanced problem within the natural training paradigm, where their ultimate goal is improving model's standard accuracy under imbalanced scenarios. However, few studies focus on improving the model's robust accuracy under the on the adversarial training paradigm[1]. In addition, as demonstrated in [102], some effective techniques for handling the imbalanced problem for the nature training paradigm are not applicable to the adversarial training paradigm. Hence, it's worthwhile to investigate new approaches to boost the model robustness under imbalanced scenarios.

Recently, data augmentation techniques have been proven to be an effective way to improve

[1] In this chapter, we denote *standard accuracy* as model's prediction accuracy on the input examples without adversarial perturbations and *robust accuracy* as model's prediction accuracy on the perturbed input examples constrained by l_∞-norm 8/255.

model robustness with respect to noisy inputs like blurred images [43, 80]. This provides a potential way to solve the imbalanced problem within the adversarial training paradigm. Therefore we propose *Imb-Mix*, a novel data augmentation based framework to advance model robustness under imbalanced scenarios. Imb-Mix first generates multiple adversarial examples for the minority classes. This is done by adding random noise to the original adversarial examples created by the PGD adversarial attack [71]. This is done to balance the imbalanced data distribution between classes. Next, to increase the generalization ability of the trained model, we construct Mixup-mimic mixed examples upon the augmented dataset used by adversarial training. Moreover, to further improve the model performance, we introduce the stochastic model weight averaging (SWA) [45] technique into our proposed framework. SWA has been shown to be effective in improving the performance of DNNs with almost no extra computational overhead. The contributions of this chapter include:

- We introduce a simple yet effective data augmentation based framework into the adversarial training paradigm to benefit adversarial training under imbalanced scenarios.
- We theoretically prove the regularization effect of our Mixup-mimic mixed examples generation technique. This provides an understanding as to why this process can be effective under imbalanced scenarios.
- We conduct extensive experiments on multiple datasets with various imbalanced scenarios to verify the effectiveness of our proposed framework.

5.2 Related Work

5.2.1 Adversarial Robustness

The existence of successful adversarial attacks [32, 11] reveals the vulnerability of DNN models. As a countermeasure against adversarial attacks, many defense methods [71, 78, 19] have been proposed to improve model robustness. Among them, adversarial training has been proven to be one of the more effective methods [4]. Generally, adversarial training aims to solve

a min-max optimization problem where the inner maximization process utilizes some existing attack methods, such as PGD [71], to generate adversarial examples that can mislead the current model mostly, and the outer minimization process enforces the model to be robust to the generated adversarial examples. Because of its effectiveness, many variants of adversarial training have been proposed to further improve the robustness of models under various settings [111, 103, 108]. More details about adversarial robustness can be obtained in related surveys [13, 109]. Note that, as the majority of existing adversarial training based methods only focus on balanced datasets. As such, the performance of these methods will be drastically decreased when the training dataset is imbalanced [106, 102].

5.2.2 Imbalanced Learning

Due to its commonality in many real-world applications [44], learning from imbalanced datasets has been widely investigated in the past few decades. Most existing works can be roughly classified into two categories, i.e., re-sampling and re-weighting. The re-sampling methods focus on balancing the data distribution through either downsizing the majority classes [7, 9] or upsizing the minority classes [46, 25, 40]. The re-weighting methods assign different weights for different classes or even different examples. For instance, Focal loss [63] allocates larger weights for wrongly-classified examples while giving smaller weights for well-classified examples based on the standard cross entropy loss. The LDAM loss [10] regularizes the minority classes more strongly than the majority classes to achieve a good generalization performance on minority classes. More details about imbalanced learning can be obtained in related surveys [41, 47]. Note that, most existing imbalanced learning methods focused on the nature training paradigm and their trained models will be crashed when facing adversarial attacks [91, 32]. Therefore, in this chapter, we present a novel framework that is able to improve the model robustness against adversarial attacks under imbalanced scenarios.

5.2.3 Data Augmentation

Data augmentation methods have been empirically shown to be an effective way of improving the generalization ability of DNN models. For instance, random flipping and cropping are two most

commonly used techniques in image classification tasks [42]. Some random occlusion techniques such as Cutout [23] can also help models to obtain better standard classification accuracy on images. Besides applying operations on every single image, Mixup [112] adopts an pair-wise linear combination of two images to create a mixed image along with a mixed corresponding label. Although simple, experimental results verify that Mixup is able to bring much better generalization performance for DNN models. Variants of Mixup have been proposed for multiple domains including, NLP [90], computer vision [97], and graphs [37, 87]. Furthermore, Mixup strategies have also been proposed to further improve the standard classification accuracy of models under different scenarios. AUGMIX [43] demonstrate that randomly mixing generated augmentations instead of original input images can improve DNN models' robustness against noisy images (e.g., blurred images). Although various kinds of data augmentation methods have been proposed, there is no existing data augmentation methods considering the problem of improving adversarial training on imbalanced data distributions.

5.3 The Proposed Framework

In this section we present our proposed framework. We first introduce the basic idea of adversarial training in Section 5.3.1. In Section 5.3.2 we present our proposed data augmentation method *Imb-Mix*. In Section 5.3.3 we introduce the stochastic model weight averaging (SWA) technique used in our framework. Lastly, in Section 5.3.4 we detail the full training algorithm of Imb-Mix.

5.3.1 Adversarial Training

In order to improve the model robustness against adversarial attacks, previous works propose to include adversarial examples generated by some adversarial attacks methods into the model training process. This helps teach the trained model to recognize adversarial examples correctly [71, 111]. Specifically, given an input example (x, y), the PGD adversarial training [71] aims to obtain a robust model f_θ where the (correct) prediction y is the same for the original sample x and the adversarial example x'. The sample x' is generated by applying an adversarially perturbation on x. The adversarial perturbations are typically bounded by a small value ϵ under L_p-norm, i.e.,

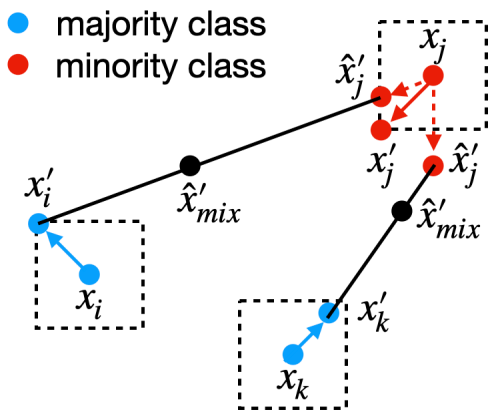


Figure 5.1 A toy example of creating augmented adversarial examples by our Imb-Mix framework. The blue and red circles represent data examples from one majority class and one minority class, respectively. The solid lines denote the process of producing adversarial examples through the inner maximization process described in Eq. (5.1). The dash lines denote the process of generating various adversarial examples for the minority class, respectively.

$\|x' - x\|_p \leq \epsilon$. Formally, the PGD adversarial training on a dataset X can be defined as,

$$\min_{\theta} \frac{1}{|X|} \sum_{i=1}^{|X|} \max_{\|x'_i - x_i\|_p \leq \epsilon} \mathcal{L}\big(f_\theta(x'_i), y_i\big). \tag{5.1}$$

Based on the PGD adversarial training, many variants have been proposed to further improve the model robustness against adversarial attacks from different aspects [111, 103]. Most existing adversarial training based methods assume that the number of training examples from each class is equally distributed. However, as pointed by a few recent works [106, 102], these methods cannot achieve satisfactory performance when the training data distribution is imbalanced.

5.3.2 Imb-Mix

To facilitate adversarial training under imbalanced scenarios, we propose a simple yet effective data augmentation based framework to balance the training data distribution. Inspired by SMOTE [15], a classical method that generates synthetic training data examples for minority classes, we focus on creating more adversarial examples for the minority classes to balance the imbalanced data distribution. This will help the model learn more useful information from minority classes, thereby improving the performance of the trained model on the those classes. Specifically, our proposed framework Imb-Mix contains two main procedures (1) supplementary adversarial example creation and (2) generated adversarial example Mixup. In the rest of this section we detail

both procedures.

5.3.2.1 Supplementary Adversarial Examples Creation

Given a data example x_i from the original imbalanced training dataset X_{org}, Imb-Mix first generates it's adversarial counterpart x_i' using the inner maximization process described in Eq. (5.1). Then, for any data example belonging to a minority class, Imb-Mix produces multiple adversarial examples based on its original adversarial example x_i' through the following the process:

$$\hat{x}_i' = x_i + \alpha \times (x_i' - x_i) \times \mathcal{N}(0, 1), \tag{5.2}$$

where α is a hyper-parameter to determine the level of perturbations added into the data example x_i. This step is repeated t times to obtain t different adversarial examples. Here the hyper-parameter t can be determined by users' domain knowledge upon the application settings. The main idea behind this design is to obtain a larger number of diverse adversarial examples for minority classes to balance the original imbalanced dataset. If a data example x_i belongs to a majority class, we set $\hat{x}_i' = x_i'$, as no additional samples are needed. After the supplementary adversarial examples creation procedure, we can obtain an augmented adversarial example set X_{adv}.

Although the aforementioned data augmentation method is able to produce t adversarial examples for every input example in the minority classes, we empirically find that the improvement of model's robustness on the minority classes is limited. We argue that this is caused by a lack of diversity between the different generated adversarial examples for each minority data example. Therefore the augmented adversarial examples produced do not contain sufficient information for the minority classes to enhance the learnt model. In addition, in many applications like fraud detection and medical diagnosis, misclassifying a minority class sample is usually more severe than misclassifying one from the majority class [68]. Therefore, we further use the idea of a re-balanced version of Mixup [112] to generate additional adversarial examples. We later show that this works as a form of regularization, thereby improving the model performance on the minority classes.

5.3.2.2 Generated Adversarial Example Mixup

To adapt Mixup into imbalanced scenarios, Remix [18] relaxes Mixup's formulation and enables the mixing factors of data and labels to be disentangled. In our Imb-Mix framework, we follow a similar idea. Specifically, for any two adversarial examples $\hat{x}'_i$ and $\hat{x}'_j$ sampled from the augmented adversarial example set X_{adv}, the mixed adversarial example $\hat{x}'_{mix}$ and it's corresponding label y_{mix} can be obtained by:

$$\begin{aligned} \hat{x}'_{mix,i} &= \lambda_x * \hat{x}'_i + (1 - \lambda_x) * \hat{x}'_j, \\ y_{mix,i} &= \lambda_y * y_i + (1 - \lambda_y) * y_j, \end{aligned} \tag{5.3}$$

where λ_x is sampled from a beta distribution $\mathcal{B}(\gamma, \gamma)$ (typically we choose $\gamma = 1.0$) and λ_y is defined as,

$$\lambda_y = \begin{cases} 0, & n_i/n_j \geq \kappa \text{ and } \lambda_x < \tau; \\ 1, & n_i/n_j \leq 1/\kappa \text{ and } 1 - \lambda_x < \tau; \\ \lambda_x, & \text{otherwise.} \end{cases} \tag{5.4}$$

Here n_i and n_j represent the number of adversarial examples for class i and class j, respectively. κ and τ are two hyper-parameters. We follow the default settings $\kappa = 3$ and $\tau = 0.5$ adopted by Remix [18] in our implementation.

By applying the aforementioned mixup procedure on all generated adversarial examples, we can obtain a set of mixed data-label pair $(\hat{x}'_{mix,i}, y_{mix,i})$, denoted as X_{mix}. Finally, the DNN classifier f_θ will be trained by minimizing the model's cross entropy loss on elements of the set X_{mix}, instead of the original imbalanced dataset X_{org}. Formally, the final objective function of our Imb-Mix framework can be described as,

$$\min_\theta \frac{1}{|X_{mix}|} \sum_{i=1}^{|X_{mix}|} l_\theta\big(f_\theta(\hat{x}'_{mix,i}), y_{mix,i}\big). \tag{5.5}$$

To better demonstrate the data augmentation process in our proposed framework Imb-Mix, we provide a toy example of applying Imb-Mix on a binary imbalanced classification problem in Figure 5.1. As shown in this example, adversarial examples for data examples x_i, x_j, and x_k will be generated first using a PGD attack [71]. Then our Imb-Mix framework will produce several different adversarial examples $\hat{x}'_j$ for the minority data example x_j by adding random noise to its

original adversarial counterpart x'_j. Finally, mixup-mimic mixed examples will be created based on $\hat{x}'_j$ and adversarial examples of the majority class $\hat{x}'_i$ and $\hat{x}'_k$,

5.3.3 Stochastic Model Weight Averaging

DNNs are typically trained by minimizing a loss function with stochastic gradient descent (SGD), which is an iterative method proposed for optimizing model parameters. Recently, some existing works [45, 5] discovered that simply averaging multiple points along the trajectory of SGD can lead to a better generalization ability. This kind of averaging strategy is called stochastic model weight averaging (SWA). Formally, it can be defined as

$$\theta_{\text{SWA}} \leftarrow \frac{\theta_{\text{SWA}} \times n_{\text{models}} + \theta}{n_{\text{models}} + 1},$$

where θ is the model parameters obtained by SGD and n_{models} is the number of models used for averaging the parameters. At the beginning of applying SWA, $\theta_{\text{SWA}} = \theta$.

In addition to the effectiveness, SWA will not add any additional costs during the model training process and can be easily integrated with any other optimization methods besides SGD. Therefore, to further improve the model robustness under imbalanced scenarios, we adopt SWA in our Imb-Mix framework. We empirically find that SWA can make a visible contribution to the performance of our Imb-Mix framework. More results can be found in Section 5.5.

5.3.4 Algorithm

The overall algorithm of our proposed framework Imb-Mix is shown in Algorithm 5.1. Given an imbalanced training dataset X_{org}, for each iteration Imb-Mix framework first obtains the augmented adversarial examples set X_{adv}. Using this, it produces the mixed adversarial example-label pairs set X_{mix} based on X_{adv}. The parameters of the DNN classifier f_θ will be updated by minimizing the model's empirical loss on X_{mix}. If the training iteration reaches a pre-defined value, then SWA will be introduced to update model parameters θ.

5.4 Regularization Effect Of Imb-Mix

In this section, we examine the properties of our proposed framework, Imb-Mix. We theoretically prove that the Mixup-mimic mixed examples generation technique adopted in Imb-Mix can

Algorithm 5.1 The algorithm of Imb-Mix.

Input: an imbalanced training dataset X_{org}
Output: a trained DNN classifier f_θ

Initialize the parameters θ of the DNN classifier f_θ.
repeat
  Obtain an augmented adversarial examples set X_{adv} based on PGD attack and Eq. (5.2).
  Get a mixed data-label pairs set X_{mix} based on Eqs. (5.3)-(5.4).
  Optimize the final objective function Eq. (5.5).
  Update the parameters $\theta \leftarrow \theta - \delta * \nabla_\theta l_\theta$.
  if swa-epochs **then**
    Apply SWA as described in Section 5.3.3.
  end if
until model convergence

be formulated as a form of regularization on the minority class examples.

To simplify our analysis, we consider a binary imbalanced setting, where only two classes are involved. Furthermore, we assume the dataset is imbalanced such that there is a majority class C_m and a minority class C_n. Recall that when performing linear interpolation on any two data examples x_i and x_j, Imb-Mix assigns two different mixing factors λ_x and λ_y for them in data space and label space, respectively. The mixing factor λ_x is sampled from a Beta distribution $\mathcal{B}(\gamma, \gamma)$. The factor λ_y is determined by the ratio between the example size of the class x_i belonging to and the example size of the class x_j belonging to, the value of λ_x and two hyper-parameters κ and τ, as shown in Eq. (5.4). More specifically, the value of λ for data examples x_i and x_j is determined by: 1) if both x_i and x_j are sampled from the majority class C_m, then $\lambda_y = \lambda_x$; 2) if both x_i and x_j are sampled from the minority class C_n, then $\lambda_y = \lambda_x$; and 3) if x_i and x_j are sampled from different classes, then λ_y can be either 0, 1 or λ_x. In our binary imbalanced case, we further assume the ratio between the example size of two classes C_m and C_n satisfy $|C_m|/|C_n| \geq \kappa$ and set $\tau = 0.5$ as adopted in Remix [18]. Hence, if x_i is sampled from the majority class C_m and x_j is sampled from the minority class C_n when $\lambda_x < 0.5$ then $\lambda_y = 0$. Otherwise we set $\lambda_y = \lambda_x$. Note that we omit the scenario where x_i is sampled from the minority class C_n and x_j is sampled from the majority class C_m, as it is equivalent to our discussed scenario. As a conclusion, λ_y can be either λ_x or 0. Next, we will analyze the regularization effect of Imb-Mix on these two cases, separately.

Case 1: $\lambda_y = 0$. In this case, Imb-Mix will only conduct linear interpolation on two data examples on the data space and then assign the label of minority class C_n to the mixed data example. Hence, the loss function on all mixed data examples, denoted as $\mathcal{L}_{Imb-Mix}(\theta)$, can be defined as

$$\mathcal{L}_{Imb-Mix}(\theta) = \frac{1}{|C_m| \times |C_n|} \sum_{i=1}^{|C_m|} \sum_{j=1}^{|C_n|} \mathbb{E}_\lambda l_\theta\Big(y_j, f_\theta\big(\lambda x_i + (1-\lambda)x_j\big)\Big). \tag{5.6}$$

where the loss function l_θ represents the binary cross-entropy loss, $y_j = C_n$ and $\lambda \sim \beta_{[0,0.5]}(\gamma, \gamma)$. For simplicity, we use λ to represent λ_x in the following.

Following [12], we define $\bar{\lambda} = \mathbb{E}_\lambda \lambda$ and introduce a random perturbation δ_i formulated as

$$\delta_i = (\lambda - \bar{\lambda})x_i + (1-\lambda)x_j - (1-\bar{\lambda})\bar{x}.$$

Then Eq. (5.6) can be rewritten as

$$\mathcal{L}_{Imb-Mix}(\theta) = \frac{1}{|C_m|} \sum_{i=1}^{|C_m|} \mathbb{E}_{\lambda,j} l_\theta\big(y_j, f_\theta(\tilde{x}_i + \delta_i)\big), \tag{5.7}$$

where $j \sim Uniform(X_{\{C_n\}})$ and $X_{\{C_n\}}$ is a set that contains all minority examples and mixed samples,

$$\tilde{x}_i = \bar{x} + \bar{\lambda}(x_i - \bar{x}).$$

For the loss function $\mathcal{L}_{Imb-Mix}(\theta)$ described in Eq. (5.7), we have the following theorem.

Theorem 5.4.1. *The Imb-Mix loss function $\mathcal{L}_{Imb-Mix}(\theta)$ defined in Eq. (5.7) can be rewritten as the following*

$$\mathcal{L}_{Imb-Mix}(\theta) = \frac{1}{|C_m|} \sum_{i=1}^{|C_m|} l_\theta\big(y_j, f_\theta(\tilde{x}_i)\big) + R1(\theta) + R2(\theta), \tag{5.8}$$

where

$$R1(\theta) = \frac{1}{2|C_m|} \sum_{i=1}^{|C_m|} \left\| \Big(\nabla f_\theta(\tilde{x}_i) - J^{(i)}\Big)^\top \Big(\nabla^2_{uu} l_\theta\big(y_j, f(\tilde{x}_i)\big)\Big)^{\frac{1}{2}} \right\|$$

$$R2(\theta) = \frac{1}{2|C_m|} \sum_{i=1}^{|C_m|} \left\langle \Sigma^{(i)}_{\tilde{x}\tilde{x}}, \nabla_u l_\theta\big(y_j, f_\theta(\tilde{x}_i)\big) \nabla^2 f_\theta(\tilde{x}_i) \right\rangle$$

and for any $i \in \{1, 2, ..., |X_{\{C_m\}}|\}$,

$$J^{(i)} = -\Big(\nabla^2_{uu} l_\theta\big(y_j, f_\theta(\tilde{x}_i)\big)\Big)^{-1} \nabla_u l_\theta\big(y_j, f_\theta(\tilde{x}_i)\big) \Sigma^{(i)}_{y_j\tilde{x}} \Big(\Sigma^{(i)}_{\tilde{x}\tilde{x}}\Big)^{-1}.$$

Proof. Inspired by recent work [12], we examine the regularizing effect of the mixing factor λ. This is achieved by approximating the loss function l_θ using a second-order quadratic Taylor approximation near each mixed example pair $(\tilde{x}_i, y_j)$. Assuming l_θ is twice differentiable and expressing the derivatives of $l(y, f(x))$ as derivatives of $l(y, u)$ and $f(x)$, then we can have

$$\begin{aligned}
&\mathbb{E}_{\lambda,j} l_\theta(y_j, f_\theta(\tilde{x}_i + \delta_i)) = l_\theta(y_j, f_\theta(\tilde{x}_i + \delta_i)) \\
&+ \frac{1}{2}\Big\langle \mathbb{E}_{\lambda,j}\delta_i\delta_i^\top, \nabla f_\theta(\tilde{x}_i)^\top \nabla^2_{uu} l_\theta(y_j, f_\theta(\tilde{x}_i))\nabla f(\tilde{x}_i) \\
&+ \nabla_u f_\theta(\tilde{x}_i)\nabla^2 f_\theta(\tilde{x}_i)\Big\rangle.
\end{aligned}$$

By replacing the expectations in the above equation by their values given by Lemma 2 in [12], we can have

$$\begin{aligned}
&\mathbb{E}_{\lambda,j} l_\theta(y_j, f_\theta(\tilde{x}_i + \delta_i)) = l_\theta(y_j, f_\theta(\tilde{x}_i)) \\
&+ \frac{1}{2}\Big\langle \Sigma^{(i)}_{\tilde{x}\tilde{x}}, \nabla f_\theta(\tilde{x}_i)^\top \nabla^2_{uu} l_\theta(y_j, f_\theta(\tilde{x}_i))\Big\rangle \\
&+ \frac{1}{2}\Big\langle \Sigma^{(i)}_{\tilde{x}\tilde{x}}, \nabla_u f_\theta(\tilde{x}_i)\nabla^2 f_\theta(\tilde{x}_i)\Big\rangle,
\end{aligned}$$

where for any $i \in \{1, 2, ..., |X_{\{C_m\}}|\}$,

$$\begin{aligned}
\Sigma^{(i)}_{\tilde{x}\tilde{x}} &= \frac{\sigma^2(\tilde{x}_i - \bar{x})(\tilde{x}_i - \bar{x})^\top + \xi^2\Sigma_{\tilde{x}\tilde{x}}}{\bar{\theta}^2}, \\
\Sigma^{(i)}_{\tilde{x}y_j} &= \frac{\xi^2\Sigma_{\tilde{x}y_j}}{\bar{\theta}^2},
\end{aligned}$$

Here $\bar{\lambda}$ and σ^2 be the mean and variance of a $\beta_{[0,0.5]}(\gamma, \gamma)$ distributed random variable, respectively, and $\xi^2 = \sigma^2 + (1 - \bar{\lambda})^2$. By summing over i, finally we can get

$$\mathcal{L}_{Imb-Mix}(\theta) = \frac{1}{|C_m|}\sum_{i=1}^{|C_m|} l_\theta\big(y_j, f_\theta(\tilde{x}_i)\big) + R1(\theta) + R2(\theta),$$

where

$$\begin{aligned}
R1(\theta) &= \frac{1}{2|C_m|}\sum_{i=1}^{|C_m|}\left\|\Big(\nabla f_\theta(\tilde{x}_i) - J^{(i)}\Big)^\top\Big(\nabla^2_{uu} l_\theta\big(y_j, f(\tilde{x}_i)\big)\Big)^{\frac{1}{2}}\right\| \\
R2(\theta) &= \frac{1}{2|C_m|}\sum_{i=1}^{|C_m|}\Big\langle \Sigma^{(i)}_{\tilde{x}\tilde{x}}, \nabla_u l_\theta\big(y_j, f_\theta(\tilde{x}_i)\big)\nabla^2 f_\theta(\tilde{x}_i)\Big\rangle
\end{aligned}$$

and for any $i \in \{1, 2, ..., |X_{\{C_m\}|}\}$,

$$J^{(i)} = -\Big(\nabla^2_{uu} l_\theta\big(y_j, f_\theta(\tilde{x}_i)\big)\Big)^{-1} \nabla_u l_\theta\big(y_j, f_\theta(\tilde{x}_i)\big) \Sigma^{(i)}_{y_j\tilde{x}} \Big(\Sigma^{(i)}_{\tilde{x}\tilde{x}}\Big)^{-1}.$$

□

As shown in Eq. (5.8), the loss function of Remix consists of three parts, $l_\theta\big(y_j, f_\theta(\tilde{x}_i)\big)$ denotes the loss value on the mixed data examples with a minority label y_j, and two additional terms $R1(\theta)$ and $R2(\theta)$. These two terms act as a regularization effect on the mixed data examples. Hence, we can see the advantages of Imb-Mix. On the one hand, Imb-Mix generates more training examples intentionally assigned to the minority class, which can help learn a better decision boundary between the majority and minority classes. This can lead to better model generalization. On the other hand, similar with Mixup, Imb-Mix can also be rewritten as an empirical risk on perturbed data examples (i.e.$(\tilde{x}_i, y_j)$). This allows us to interpret Imb-Mix as a form of regularization. The regularization helps the model avoid simply remembering data examples in the original training dataset and improve the generalization ability of the model.

Case 2: $\lambda_y = \lambda_x$. In this case, the Imb-Mix performs exactly same with Mixup [112]. Hence, based on [12], we have

$$\begin{aligned} &\mathcal{L}_{Imb-Mix}(\theta) = \mathcal{L}_{Mixup}(\theta) \\ &= \frac{1}{W} \sum_{i=1}^{W} l_\theta\big(\tilde{y}_i, f_\theta(\tilde{x}_i)\big) + R1(\theta) + R2(\theta) + R3(\theta) + R4(\theta). \end{aligned} \tag{5.9}$$

Here we use W to denote the number of example pairs used in Mixup and

$$\tilde{y}_i = \bar{y} + \bar{\lambda}(y_i - \bar{y}).$$

Similarly, there are four regularization terms, i.e., $R1(\theta)$, $R2(\theta)$, $R3(\theta)$ and $R4(\theta)$, in the loss function of Mixup, which can effectively improve the generalization ability of the trained model. For details of these four regularization terms, please refer to [12].

5.5 Experiment

In this section, we conduct various experiments to validate the effectiveness of our Imb-Mix framework. We aim at answering the following two questions:

- Can the proposed framework Imb-Mix boost adversarial training under various imbalanced scenarios?
- What is the impact of each component on Imb-Mix?

We begin by introducing the experimental settings including datasets construction and implementation details. Next, we compare Imb-Mix with several representative methods to answer the first question. Then we analyze the impact of each component on Imb-Mix to answer the second question.

5.5.1 Experimental Settings

Datasets. We create several imbalanced training datasets based on two benchmark image datasets CIFAR10 and CIFAR100 [56] with diverse imbalanced distributions. Specifically, following existing imbalanced learning works, we consider two different imbalance types: Exponential (Exp) imbalance [21] and Step imbalance [7]. For Exp imbalance, the number of training examples of each class will be reduced according to an exponential function $n = n_i \tau^i$, where i is the class index, n_i is the number of training data examples in the original training dataset for class i and $\tau \in (0, 1)$. We categorize the half of the classes with most frequent example sizes in the imbalanced training dataset as majority classes and the remaining half as minority classes. For Step imbalance, we equally split the classes into majority and minority classes where the number of training data examples are equal in each majority/minority class. Moreover, we denote *imbalance ratio* K as the ratio between training example sizes of the most frequent and least frequent classes. We construct different imbalanced datasets "Step-10", "Step-100", "Exp-10" and "Exp-100", by adopting different imbalanced types (Step or Exp) with different imbalanced ratios ($K = 10$ or $K = 100$) to train models. We evaluate the model's performance on the original uniformly distributed test datasets of CIFAR10 and CIFAR100 correspondingly.

Baseline methods. We implement several representative imbalanced learning methods (or their combinations) into adversarial training as baseline methods. These methods include: (1) the original PGD adversarial training (vanilla adv.); (2) PGD adversarial training with re-sample (adv.

Table 5.1 Experiment results on the CIFAR10 Exp-10 dataset under l_∞ threat model.

Metric	Standard Accuracy		Robust Accuracy	
Method	Overall	Minority	Overall	Minority
vanilla adv.	72.42 ± 0.52	64.15 ± 0.55	33.18 ± 0.60	22.57 ± 0.60
adv. + resample	68.72 ± 0.97	57.57 ± 1.86	31.68 ± 0.19	20.90 ± 0.77
adv. + reweight	72.91 ± 0.26	65.50 ± 0.24	33.22 ± 0.40	24.29 ± 0.53
mixup + adv.	75.26 ± 0.43	67.91 ± 0.26	37.23 ± 0.15	26.23 ± 0.08
remix + adv.	75.61 ± 0.60	69.34 ± 0.91	37.39 ± 0.54	27.45 ± 0.75
Imb-Mix	**78.17 ± 0.18**	**73.03 ± 0.44**	**39.00 ± 0.27**	**30.52 ± 0.37**

Table 5.2 Experiment results on the CIFAR10 Exp-100 dataset under l_∞ threat model.

Metric	Standard Accuracy		Robust Accuracy	
Method	Overall	Minority	Overall	Minority
vanilla adv.	49.30 ± 1.13	23.40 ± 1.71	24.29 ± 0.38	4.67 ± 0.18
adv. + resample	45.61 ± 0.44	18.38 ± 1.86	23.12 ± 0.32	4.41 ± 0.82
adv. + reweight	50.89 ± 0.69	25.69 ± 1.26	24.35 ± 0.47	5.73 ± 0.42
mixup + adv.	50.33 ± 1.22	25.67 ± 1.75	26.47 ± 0.29	5.47 ± 0.55
remix + adv.	51.35 ± 1.45	27.13 ± 2.07	26.64 ± 0.27	6.18 ± 0.76
Imb-Mix	**55.15 ± 0.42**	**32.93 ± 0.63**	**26.87 ± 0.34**	**7.95 ± 0.55**

Table 5.3 Experiment results on the CIFAR10 Step-10 dataset under l_∞ threat model.

Metric	Standard Accuracy		Robust Accuracy	
Method	Overall	Minority	Overall	Minority
vanilla adv.	66.09 ± 0.25	45.07 ± 0.50	32.71 ± 0.40	12.81 ± 0.29
adv. + resample	59.90 ± 0.77	34.81 ± 0.96	31.92 ± 0.17	10.29 ± 0.47
adv. + reweight	67.47 ± 0.14	48.71 ± 0.22	33.12 ± 0.16	14.90 ± 0.06
mixup + adv.	66.99 ± 1.34	46.20 ± 2.28	36.08 ± 0.15	14.17 ± 1.17
remix + adv.	67.78 ± 1.07	47.91 ± 1.65	36.37 ± 0.13	14.86 ± 0.62
Imb-Mix	**71.70 ± 0.18**	**55.31 ± 0.30**	**37.89 ± 0.20**	**18.49 ± 0.45**

+ resample), where the probability of each example to be selected in each training batch equals to the inverse of the effective number of each class; (3) PGD adversarial training with reweighting (adv. + reweight), where each example is reweighted proportionally by the inverse of the effective number of its class; (4) PGD adversarial training with Mixup (mixup + adv.), where we apply Mixup [112] on pair-wise adversarial examples generated by PGD attack; and (5) PGD adversarial training with Remix [18] (remix + adv.), where we apply Remix on pair-wise adversarial examples generated by PGD attack.

Implementation details. All aforementioned methods are implemented using a Pytorch library DeepRobust [62]. For CIFAR10 and CIFAR100 based datasets, the adversarial examples used in

Table 5.4 Experiment results on the CIFAR10 Step-100 dataset under l_∞ threat model.

Metric	Standard Accuracy		Robust Accuracy	
Method	Overall	Minority	Overall	Minority
vanilla adv.	47.59 ± 0.40	7.61 ± 1.06	28.14 ± 0.16	0.82 ± 0.11
adv. + resample	43.49 ± 0.08	2.97 ± 0.50	28.75 ± 0.45	0.50 ± 0.17
adv. + reweight	46.98 ± 0.42	9.44 ± 0.25	28.47 ± 0.12	**1.38 ± 0.16**
mixup + adv.	44.58 ± 0.74	2.27 ± 1.55	29.61 ± 0.66	0.19 ± 0.17
remix + adv.	47.08 ± 1.19	7.24 ± 2.09	29.69 ± 0.61	0.81 ± 0.37
Imb-Mix	**48.05 ± 0.20**	**9.81 ± 0.46**	**30.25 ± 0.20**	1.29 ± 0.28

Table 5.5 Experiment results on the CIFAR100 Exp-10 dataset under l_∞ threat model.

Metric	Standard Accuracy		Robust Accuracy	
Method	Overall	Minority	Overall	Minority
vanilla adv.	41.36 ± 0.25	30.84 ± 0.20	15.06 ± 0.15	9.95 ± 0.09
adv. + resample	38.49 ± 0.18	25.88 ± 0.30	14.81 ± 0.07	9.49 ± 0.34
adv. + reweight	39.85 ± 0.40	28.23 ± 0.39	14.70 ± 0.04	9.59 ± 0.21
mixup + adv.	46.39 ± 0.29	34.21 ± 0.39	18.84 ± 0.21	12.35 ± 0.21
remix + adv.	47.01 ± 0.37	35.77 ± 0.33	18.82 ± 0.25	12.96 ± 0.28
Imb-Mix	**51.22 ± 0.49**	**41.32 ± 1.00**	**19.14 ± 0.21**	**13.99 ± 0.41**

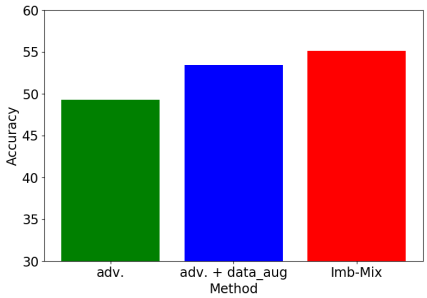
Table 5.6 Experiment results on the CIFAR100 Step-10 dataset under l_∞ threat model.

Metric	Standard Accuracy		Robust Accuracy	
Method	Overall	Minority	Overall	Minority
vanilla adv.	39.68 ± 0.33	18.46 ± 0.24	15.43 ± 0.07	5.13 ± 0.30
adv. + resample	36.16 ± 0.39	12.07 ± 0.44	15.51 ± 0.13	4.14 ± 0.31
adv. + reweight	38.33 ± 0.47	15.73 ± 0.60	15.12 ± 0.13	4.78 ± 0.24
mixup + adv.	42.62 ± 0.28	17.28 ± 0.61	19.35 ± 0.17	5.77 ± 0.28
remix + adv.	43.90 ± 0.17	20.59 ± 0.42	19.60 ± 0.10	6.51 ± 0.35
Imb-Mix	**46.79 ± 0.67**	**26.33 ± 1.13**	**19.94 ± 0.31**	**7.84 ± 0.52**

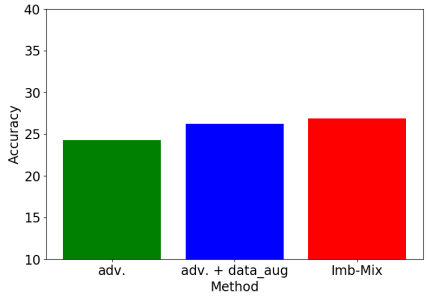
training are calculated by PGD-10, with a perturbation budget $\epsilon = 8/255$ and step size $\gamma = 2/255$. In evaluation, we report robust accuracy under l_∞-norm $8/255$ attacks generated by PGD-20 on Resnet-18 [42] models. We set the total training epochs to 250 and the initial learning rate to 0.1, and decay the learning rate at epoch 160 and 180 with the ratio 0.01.

5.5.2 Performance Comparison

Tables 5.1-5.6 report the performance comparison on multiple imbalanced datasets with various imbalanced scenarios. The highest accuracy achieved among all methods are denoted by bold values. From these tables, we have the following observations. First, compared to baseline methods, Imb-Mix obtains improved performance in terms of both overall standard accuracy and

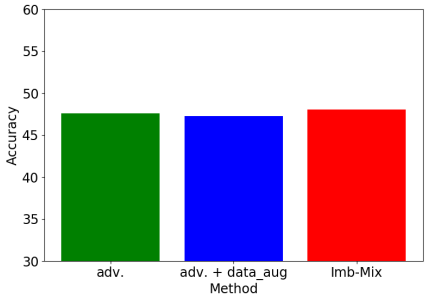


(a) Standard Accuracy

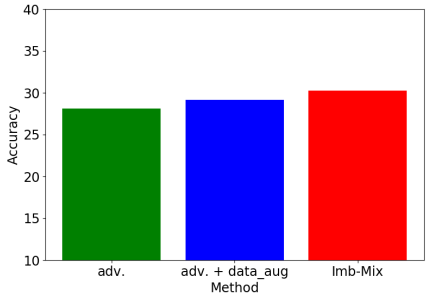


(b) Robust Accuracy

Figure 5.2 Performance on the CIFAR10 Exp-100 dataset.



(a) Standard Accuracy



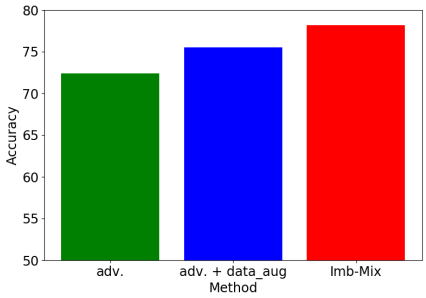
(b) Robust Accuracy

Figure 5.3 Performance on the CIFAR10 Step-100 dataset.

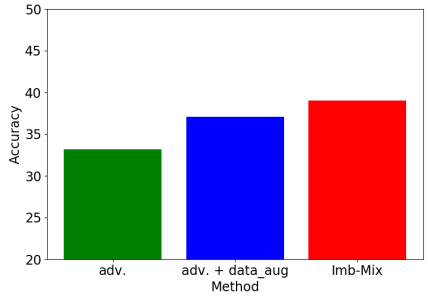
robust accuracy under almost all imbalanced scenarios. This suggests that Imb-Mix is able to facilitate adversarial training under imbalanced scenarios. Second, Imb-Mix obtains significant improvement on those under-represented classes with a large margin. For instance, on the CIFAR10 Exp-10 dataset, Imb-Mix improves the standard accuracy on minority classes from 69.34% achieved by the best baseline method to 73.03% and robust accuracy from 27.45% to 30.52%. These results demonstrate that Imb-Mix is able to obtain more robustness under imbalanced settings. In addition, we find that the baseline method *adv.* + *re-sample* always achieves the worst performance among all methods. This demonstrates that simply combining adversarial training with re-sampling techniques cannot improve the models' robustness under imbalanced scenarios. In other words, novel data augmentation methods, such as our proposed framework Imb-Mix, are necessary.

5.5.3 Ablation Studies

In this subsection, we investigate how each component contributes to Imb-Mix. This includes our proposed data augmentation method as well as the SWA technique. We further explore the

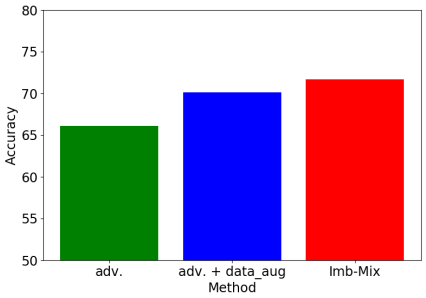


(a) Standard Accuracy

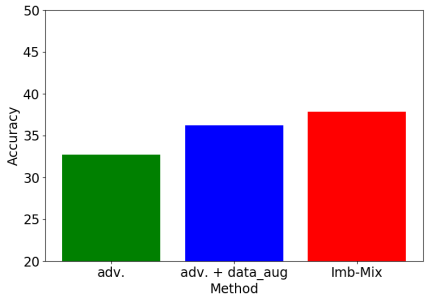


(b) Robust Accuracy

Figure 5.4 Performance on the CIFAR10 Exp-10 dataset.



(a) Standard Accuracy

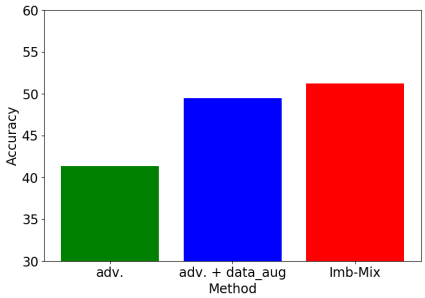


(b) Robust Accuracy

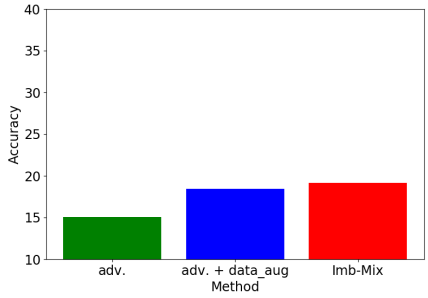
Figure 5.5 Performance on the CIFAR10 Step-10 dataset.

performance under various imbalanced scenarios. To achieve this goal, we first implemented a variant of Imb-Mix, which only integrates our proposed data augmentation method with adversarial training and adopts stochastic gradient descent (SGD) to optimize the loss function. We then compared the performance of this method, i.e. adv. + data_aug, with vanilla adversarial training and our Imb-Mix framework on our constructed imbalanced datasets.

Figures 5.2-5.7 show both standard accuracy and robust accuracy achieved by aforementioned three methods on different imbalanced datasets. From these figures, we can have the following observations. First, our proposed data augmentation method indeed benefits adversarial training under imbalanced scenarios. Compared to vanilla adversarial training, adversarial training combined with our data augmentation method achieves significant improvement on both clean examples and adversarial examples. For example, on CIFAR100 Step-10 datasets, adversarial training combined with our data augmentation method obtains a standard accuracy of 50% and a robust accuracy 20%. However, vanilla adversarial training only achieves a 40% and 15% standard accuracy and

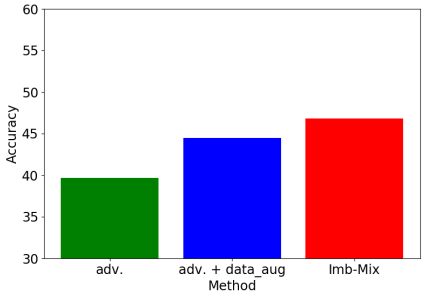


(a) Standard Accuracy

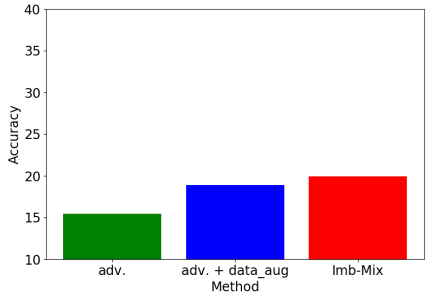


(b) Robust Accuracy

Figure 5.6 Performance on the CIFAR100 Exp-10 dataset.



(a) Standard Accuracy



(b) Robust Accuracy

Figure 5.7 Performance on the CIFAR100 Step-10 dataset.

robust accuracy, respectively. Secondly, our Imb-Mix framework achieves the best performance among three methods. This verifies the effectiveness of the SWA technique adopted by our Imb-Mix framework, as the only difference between Imb-Mix and adv. + data_aug is that the former applies SWA while the latter utilizes the normal SGD during the model training process. To sum up, experimental results reported in Figures 5.2-5.7 demonstrate the contribution of each component to our framework.

5.5.4 Robustness against l_2 Attack

To further evaluate the effectiveness of our Imb-Mix framework, we also adversarially train Resnet-18 [42] models on CIFAR100 Exp-10 dataset under l_2 attack. We follow the same settings as in [105] with s perturbation budget of $\epsilon = 128/255$ and a step size of $\gamma = 15/255$. As shown in Table 5.7, Imb-Mix outperforms all baseline methods with a large margin. This further verifies the effectiveness of Imb-Mix.

Table 5.7 Experiment results on the CIFAR100 Exp-10 dataset under l_2 threat model.

Metric	Standard Accuracy		Robust Accuracy	
Method	Overall	Minority	Overall	Minority
vanilla adv.	58.17 ± 0.07	47.81 ± 0.29	47.22 ± 0.31	37.81 ± 0.76
adv. + resample	55.77 ± 0.22	44.25 ± 0.73	45.84 ± 0.36	35.18 ± 0.71
adv. + reweight	57.70 ± 0.22	47.43 ± 0.46	47.13 ± 0.33	37.87 ± 0.75
mixup + adv.	62.60 ± 0.06	51.45 ± 0.16	52.47 ± 0.47	41.35 ± 0.37
remix + adv.	63.22 ± 0.44	53.49 ± 0.66	52.67 ± 0.49	42.97 ± 0.79
Imb-Mix	**63.97 ± 0.54**	**54.05 ± 0.97**	**53.51 ± 0.16**	**43.38 ± 0.47**

5.6 Chapter Conclusion

In this chapter, we propose a novel data augmentation based framework, *Imb-Mix*, to facilitate the adversarial training method under imbalanced scenarios. Imb-Mix first generates adversarial examples for the minority classes to balance the dataset. It then constructs Mixup-mimic mixed examples as inputs during the model training process. In addition, stochastic model weight averaging is also included in our framework and helps achieve better performance. We validate the effectiveness of Imb-Mix via comprehensive experiments. In the future, we plan to investigate more advanced data augmentation methods to further improve the model robustness under imbalanced scenarios.

CHAPTER 6

CONCLUSIONS

In this chapter, I summarize the research efforts described in this dissertation and discuss promising research directions.

6.1 Dissertation Summary

In this dissertation, I introduced my studies on learning from imbalanced data distribution under various kind of settings. Specifically, I presented several effective solutions for (1) generating high-quality synthetic data to balance data distribution, (2) learning from imbalanced crowdsourced labeled data, and (3) improving model robustness given imbalanced training data.

To generate more realistic realistic and discriminative data samples for minority classes, in Chapter 2, I first pointed out the importance of both local and global data distribution information in generating high-quality synthetic minority samples to tackle the class imbalance problem. Based on that, I proposed GL-GAN [100], a novel data generation framework utilizing both global and local information of the given imbalanced data in the synthetic minority sample generation process. Comparing with common related works only considers local data distribution information when generating synthetic minority samples, as shown in experimental results, GL-GAN is able to produce more realistic and discriminative synthetic minority samples by taking global data distribution information into consideration.

To learn useful information from imbalanced crowdsourced labeled data, in Chapter 3, I proposed a deep neural network based classifier ICED [99]. During training, a true label inference module equipped in ICED will estimate determinate true labels from given crowdsourced labeled data by the true label inference module while a synthetic data generation module will generate synthetic data samples for the minority class using the estimated determinate true labels. These two modules are able to augment each other and improve themselves iteratively. With the help of these modules, ICED is able to infers true labels from imbalanced crowdsourced labeled data and achieves high accuracy on the classification task simultaneously. I conducted a series of experiments to verify the effectiveness of ICED.

To improving model robustness under imbalanced scenarios, I explored several solutions from different perspectives. In Chapter 4, I demonstrated that adversarial training alone cannot be effective for improving the robustness of models under imbalanced scenarios, because of adversarially trained models can suffer much worse performance on minority classes observed in empirically studies, and simply combing adversarial training with reweighting strategies also cannot work well, due to the poor data separability brought by adversarial training training proven by theoretical analysis. Based on findings, I proposed a novel method SRAT [102] to boost the reweighting strategy in adversarial training under imbalanced scenarios. By testing the performance of SRAT in various kinds of experiments, I validated the effectiveness of it. In Chapter 5, I focused on boosting adversarial training under imbalanced scenarios by augmenting imbalanced training data. The proposed framework Imb-Mix [98] is able to generate multiple adversarial examples for minority classes, by adding random noise to the original adversarial examples created by one specific adversarial attack method first and then constructing Mixup-mimic mixed examples upon the augmented dataset used by adversarial training. I also theoretically proven the regularization effect of the Mixup-mimic mixed examples generation technique adopted in Imb-Mix. Experimental results demonstrated that data augmentation can also be an effective way to benefit adversarial training under imbalanced scenarios.

6.2 Future Work

In addition to the achievements obtained by my studies, I also plan to explore the following research directions in the future:

- **Multi-label Imbalanced Classification.** Multi-label classification task is omnipresent in many real-world applications, such as annotating a given movie category and creating a profile for a customer. Different from the common multi-class classification task I investigated in this dissertation, in multi-label classification task, each data sample is typically associated with series of labels instead of one and there is no constraint on how many labels one data sample can be assigned to. Hence, the imbalanced data distribution as almost unavoidable in the multi-label classification task, as it's very hard to guarantee each label occur with

the same number. I plan to explore how to learn from imbalanced multi-label data more effectively to obtain satisfied performance on the multi-label classification task.

- **Learning from Imbalanced Text Data.** Most existing approaches for handling imbalanced data distribution mainly focused on continuous data like image, and research on addressing this problem on non-continuous data, such as text, is rather limited. Considering that text data is everywhere in human society, this direction deserves more attention. Hence, as one future work, I plan to investigate the negative impacts brought by the imbalanced data distribution in various text data related applications, such as text classification and sentiment analysis, and explore effective solutions to mitigate negative impacts.

BIBLIOGRAPHY

[1] Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, Li Deng, Gerald Penn, and Dong Yu. Convolutional neural networks for speech recognition. *IEEE/ACM Transactions on audio, speech, and language processing*, 22(10):1533–1545, 2014.

[2] Lida Abdi and Sattar Hashemi. To combat multi-class imbalanced problems by means of over-sampling techniques. *IEEE Transactions on Knowledge and Data Engineering*, (1):238–251, 2016.

[3] Shadi Albarqouni, Christoph Baur, Felix Achilles, Vasileios Belagiannis, Stefanie Demirci, and Nassir Navab. Aggnet: deep learning from crowds for mitosis detection in breast cancer histology images. *IEEE transactions on medical imaging*, 35(5):1313–1321, 2016.

[4] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning*, pages 274–283. PMLR, 2018.

[5] Ben Athiwaratkun, Marc Finzi, Pavel Izmailov, and Andrew Gordon Wilson. There are many consistent explanations of unlabeled data: Why you should average. *arXiv preprint arXiv:1806.05594*, 2018.

[6] Jesús Bobadilla, Fernando Ortega, Antonio Hernando, and Abraham Gutiérrez. Recommender systems survey. *Knowledge-based systems*, 46:109–132, 2013.

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

[8] Lukas Budach, Moritz Feuerpfeil, Nina Ihde, Andrea Nathansen, Nele Noack, Hendrik Patzlaff, Hazar Harmouch, and Felix Naumann. The effects of data quality on machine learning performance. *arXiv preprint arXiv:2207.14529*, 2022.

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

[10] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. In *Advances in Neural Information Processing Systems*, pages 1567–1578, 2019.

[11] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017.

[12] Luigi Carratino, Moustapha Cissé, Rodolphe Jenatton, and Jean-Philippe Vert. On mixup regularization. *arXiv preprint arXiv:2006.06049*, 2020.

[13] Anirban Chakraborty, Manaar Alam, Vishal Dey, Anupam Chattopadhyay, and Debdeep Mukhopadhyay. Adversarial attacks and defences: A survey. *arXiv preprint arXiv:1810.00069*, 2018.

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

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

[16] Nitesh V Chawla, Nathalie Japkowicz, and Aleksander Kotcz. Special issue on learning from imbalanced data sets. *ACM SIGKDD explorations newsletter*, 6(1):1–6, 2004.

[17] Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. In *Proceedings of the IEEE international conference on computer vision*, pages 2722–2730, 2015.

[18] Hsin-Ping Chou, Shih-Chieh Chang, Jia-Yu Pan, Wei Wei, and Da-Cheng Juan. Remix: rebalanced mixup. In *European Conference on Computer Vision*, pages 95–110. Springer, 2020.

[19] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, pages 1310–1320. PMLR, 2019.

[20] Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A Bharath. Generative adversarial networks: An overview. *IEEE Signal Processing Magazine*, 35(1):53–65, 2018.

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

[22] Alexander Philip Dawid and Allan M Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 28(1):20–28, 1979.

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

[24] Georgios Douzas and Fernando Bacao. Effective data generation for imbalanced learning using conditional generative adversarial networks. *Expert Systems with applications*, 91:464–471, 2018.

[25] Chris Drummond, Robert C Holte, et al. C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In *Workshop on learning from imbalanced datasets II*, volume 11, pages 1–8. Citeseer, 2003.

[26] Andrew Estabrooks, Taeho Jo, and Nathalie Japkowicz. A multiple resampling method for learning from imbalanced data sets. *Computational intelligence*, 20(1):18–36, 2004.

[27] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[28] Ju Fan, Guoliang Li, Beng Chin Ooi, Kian-lee Tan, and Jianhua Feng. icrowd: An adaptive crowdsourcing framework. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pages 1015–1030, 2015.

[29] Zhe Gan, Liqun Chen, Weiyao Wang, Yuchen Pu, Yizhe Zhang, Hao Liu, Chunyuan Li, and Lawrence Carin. Triangle generative adversarial networks. In *Advances in Neural Information Processing Systems*, pages 5247–5256, 2017.

[30] Vaishali Ganganwar. An overview of classification algorithms for imbalanced datasets. *International Journal of Emerging Technology and Advanced Engineering*, 2(4):42–47, 2012.

[31] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[32] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[33] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. Ieee, 2013.

[34] Melody Guan, Varun Gulshan, Andrew Dai, and Geoffrey Hinton. Who said what: Modeling individual labelers improves classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 3109–3118, 2018.

[35] Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. Borderline-smote: a new over-sampling method in imbalanced data sets learning. In *International Conference on Intelligent Computing*, pages 878–887. Springer, 2005.

[36] Jiangfan Han, Ping Luo, and Xiaogang Wang. Deep self-learning from noisy labels. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5138–5147, 2019.

[37] Xiaotian Han, Zhimeng Jiang, Ninghao Liu, and Xia Hu. G-mixup: Graph data augmentation for graph classification. In *International Conference on Machine Learning*, pages 8230–8248. PMLR, 2022.

[38] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, et al. Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*, 2014.

[39] Haibo He, Yang Bai, Edwardo A Garcia, and Shutao Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *Neural Networks. IJCNN. IEEE International Joint Conference on*, pages 1322–1328. IEEE, 2008.

[40] Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9):1263–1284, 2009.

[41] Haibo He and Yunqian Ma. Imbalanced learning: foundations, algorithms, and applications. 2013.

[42] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[43] Dan Hendrycks, Norman Mu, Ekin D Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. *arXiv preprint arXiv:1912.02781*, 2019.

[44] Yueh-Min Huang, Chun-Min Hung, and Hewijin Christine Jiau. Evaluation of neural networks and data mining methods on a credit assessment task for class imbalance problem. *Nonlinear Analysis: Real World Applications*, 7(4):720–747, 2006.

[45] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

[46] Nathalie Japkowicz and Shaju Stephen. The class imbalance problem: A systematic study. *Intelligent data analysis*, 6(5):429–449, 2002.

[47] Justin M Johnson and Taghi M Khoshgoftaar. Survey on deep learning with class imbalance. *Journal of Big Data*, 6(1):1–54, 2019.

[48] Hiroshi Kajino, Yuta Tsuboi, and Hisashi Kashima. Clustering crowds. In *Proceedings of the twenty-seventh AAAI conference on artificial intelligence*, pages 1120–1127, 2013.

[49] Taskin Kavzoglu. Increasing the accuracy of neural network classification using refined training data. *Environmental Modelling & Software*, 24(7):850–858, 2009.

[50] Ashish Khetan, Zachary C Lipton, and Anima Anandkumar. Learning from noisy singly-labeled data. *arXiv preprint arXiv:1712.04577*, 2017.

[51] Ashish Khetan, Zachary C Lipton, and Animashree Anandkumar. Learning from noisy singly-labeled data. In *International Conference on Learning Representations*, 2018.

[52] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362*, 2020.

[53] Yassin Kortli, Maher Jridi, Ayman Al Falou, and Mohamed Atri. Face recognition systems: A survey. *Sensors*, 20(2):342, 2020.

[54] György Kovács. smote-variants: a python implementation of 85 minority oversampling techniques. *Neurocomputing*, 2019.

[55] Bartosz Krawczyk. Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4):221–232, 2016.

[56] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

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

[58] Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

[59] Kuang-Huei Lee, Xiaodong He, Lei Zhang, and Linjun Yang. Cleannet: Transfer learning for scalable image classifier training with label noise. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5447–5456, 2018.

[60] 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.

[61] Junnan Li, Yongkang Wong, Qi Zhao, and Mohan S Kankanhalli. Learning to learn from noisy labeled data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5051–5059, 2019.

[62] Yaxin Li, Wei Jin, Han Xu, and Jiliang Tang. Deeprobust: A pytorch library for adversarial attacks and defenses. *arXiv preprint arXiv:2005.06149*, 2020.

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

[64] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[65] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen Awm Van Der Laak, Bram Van Ginneken, and Clara I Sánchez. A survey on deep learning in medical image analysis. *Medical image analysis*, 42:60–88, 2017.

[66] Alexander Liu, Joydeep Ghosh, and Cheryl E Martin. Generative oversampling for mining imbalanced datasets. In *DMIN*, pages 66–72, 2007.

[67] Xu-Ying Liu, Jianxin Wu, and Zhi-Hua Zhou. Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(2):539–550, 2008.

[68] Xu-Ying Liu, Jianxin Wu, and Zhi-Hua Zhou. Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(2):539–550, 2009.

[69] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2537–2546, 2019.

[70] Rushi Longadge and Snehalata Dongre. Class imbalance problem in data mining review. *arXiv preprint arXiv:1305.1707*, 2013.

[71] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[72] Giovanni Mariani, Florian Scheidegger, Roxana Istrate, Costas Bekas, and Cristiano Malossi. Bagan: Data augmentation with balancing gan. *arXiv preprint arXiv:1803.09655*, 2018.

[73] Brian W Matthews. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2):442–451, 1975.

[74] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[75] Iqbal Muhammad and Zhu Yan. Supervised machine learning approaches: A survey. *ICTACT Journal on Soft Computing*, 5(3), 2015.

[76] Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with noisy labels. In *Advances in neural information processing systems*, pages 1196–1204, 2013.

[77] William S Noble. What is a support vector machine? *Nature biotechnology*, 24(12):1565–1567, 2006.

[78] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. *arXiv preprint arXiv:1801.09344*, 2018.

[79] Vikas C Raykar, Shipeng Yu, Linda H Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. Learning from crowds. *Journal of Machine Learning Research*, 11(4), 2010.

[80] Sylvestre-Alvise Rebuffi, Sven Gowal, Dan Andrei Calian, Florian Stimberg, Olivia Wiles, and Timothy A Mann. Data augmentation can improve robustness. *Advances in Neural Information Processing Systems*, 34:29935–29948, 2021.

[81] William Rivera. Noise reduction a priori synthetic over-sampling for class imbalanced data sets. *Information Sciences*, 408:146–161, 10 2017.

[82] Filipe Rodrigues and Francisco Pereira. Deep learning from crowds. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 1611–1618, 2018.

[83] Filipe Rodrigues, Francisco Pereira, and Bernardete Ribeiro. Gaussian process classification and active learning with multiple annotators. In *International conference on machine learning*, pages 433–441, 2014.

[84] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

[85] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. *arXiv preprint arXiv:1804.11285*, 2018.

[86] Shiven Sharma, Colin Bellinger, Bartosz Krawczyk, Osmar Zaiane, and Nathalie Japkowicz. Synthetic oversampling with the majority class: A new perspective on handling extreme imbalance. In *2018 IEEE International Conference on Data Mining*, pages 447–456. IEEE, 2018.

[87] Harry Shomer, Wei Jin, Wentao Wang, and Jiliang Tang. Toward degree bias in embedding-based knowledge graph completion. In *Proceedings of the ACM Web Conference 2023*, pages 705–715, 2023.

[88] Kai Shu, Amy Sliva, Suhang Wang, Jiliang Tang, and Huan Liu. Fake news detection on social media: A data mining perspective. *ACM SIGKDD Explorations Newsletter*, 19(1):22–36, 2017.

[89] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[90] Lichao Sun, Congying Xia, Wenpeng Yin, Tingting Liang, S Yu Philip, and Lifang He. Mixup-transformer: Dynamic data augmentation for nlp tasks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3436–3440, 2020.

[91] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[92] Ryutaro Tanno, Ardavan Saeedi, Swami Sankaranarayanan, Daniel C Alexander, and Nathan Silberman. Learning from noisy labels by regularized estimation of annotator confusion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11244–11253, 2019.

[93] Tatiana Tommasi, Francesco Orabona, and Barbara Caputo. Discriminative cue integration for medical image annotation. *Pattern Recognition Letters*, 29(15):1996–2002, 2008.

[94] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[95] Grant Van Horn and Pietro Perona. The devil is in the tails: Fine-grained classification in the wild. *arXiv preprint arXiv:1709.01450*, 2017.

[96] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[97] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *International conference on machine learning*, pages 6438–6447. PMLR, 2019.

[98] Wentao Wang, Harry Shomer, Yuxuan Wan, Yaxin Li, Jiangtao Huang, and Hui Liu. A mix-up strategy to enhance adversarial training with imbalanced data. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 2637–2645, 2023.

[99] Wentao Wang, Joseph Thekinen, Xiaorui Liu, Zitao Liu, and Jiliang Tang. Learning from imbalanced crowdsourced labeled data. In *Proceedings of the 2022 SIAM International Conference on Data Mining (SDM)*, pages 594–602. SIAM, 2022.

[100] Wentao Wang, Suhang Wang, Wenqi Fan, Zitao Liu, and Jiliang Tang. Global-and-local aware data generation for the class imbalance problem. In *Proceedings of the 2020 SIAM International Conference on Data Mining*, pages 307–315. SIAM, 2020.

[101] Wentao Wang, Guowei Xu, Wenbiao Ding, Yan Huang, Guoliang Li, Jiliang Tang, and Zitao Liu. Representation learning from limited educational data with crowdsourced labels. *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[102] Wentao Wang, Han Xu, Xiaorui Liu, Yaxin Li, Bhavani Thuraisingham, and Jiliang Tang. Imbalanced adversarial training with reweighting. In *2022 IEEE International Conference on Data Mining (ICDM)*, pages 1209–1214. IEEE, 2022.

[103] Yisen Wang, Difan Zou, Jinfeng Yi, James Bailey, Xingjun Ma, and Quanquan Gu. Improving adversarial robustness requires revisiting misclassified examples. In *International Conference on Learning Representations*, 2019.

[104] Jacob Whitehill, Ting-fan Wu, Jacob Bergsma, Javier R Movellan, and Paul L Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in neural information processing systems*, pages 2035–2043, 2009.

[105] Dongxian Wu, Shu-Tao Xia, and Yisen Wang. Adversarial weight perturbation helps robust generalization. *Advances in Neural Information Processing Systems*, 33, 2020.

[106] Tong Wu, Ziwei Liu, Qingqiu Huang, Yu Wang, and Dahua Lin. Adversarial robustness under long-tailed distribution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8659–8668, 2021.

[107] Da Xu, Yuting Ye, and Chuanwei Ruan. Understanding the role of importance weighting for deep learning. *arXiv preprint arXiv:2103.15209*, 2021.

[108] Han Xu, Xiaorui Liu, Yaxin Li, and Jiliang Tang. To be robust or to be fair: Towards fairness in adversarial training. *arXiv preprint arXiv:2010.06121*, 2020.

[109] Han Xu, Yao Ma, Hao-Chen Liu, Debayan Deb, Hui Liu, Ji-Liang Tang, and Anil K Jain. Adversarial attacks and defenses in images, graphs and text: A review. *International Journal of Automation and Computing*, 17(2):151–178, 2020.

[110] Show-Jane Yen and Yue-Shi Lee. Under-sampling approaches for improving prediction of the minority class in an imbalanced dataset. In *Intelligent Control and Automation*, pages 731–740. Springer, 2006.

[111] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*, pages 7472–7482. PMLR, 2019.

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

[113] Yudian Zheng, Guoliang Li, Yuanbing Li, Caihua Shan, and Reynold Cheng. Truth inference in crowdsourcing: Is the problem solved? *Proceedings of the VLDB Endowment*, 10(5):541–552, 2017.