

June 2023

## Deep Learning Enhancement and Privacy-Preserving Deep Learning: A Data-Centric Approach

Hung S. Nguyen  
*University of South Florida*

Follow this and additional works at: <https://digitalcommons.usf.edu/etd>



Part of the [Artificial Intelligence and Robotics Commons](#), and the [Electrical and Computer Engineering Commons](#)

---

### Scholar Commons Citation

Nguyen, Hung S., "Deep Learning Enhancement and Privacy-Preserving Deep Learning: A Data-Centric Approach" (2023). *USF Tampa Graduate Theses and Dissertations*.  
<https://digitalcommons.usf.edu/etd/9989>

This Dissertation is brought to you for free and open access by the USF Graduate Theses and Dissertations at Digital Commons @ University of South Florida. It has been accepted for inclusion in USF Tampa Graduate Theses and Dissertations by an authorized administrator of Digital Commons @ University of South Florida. For more information, please contact [digitalcommons@usf.edu](mailto:digitalcommons@usf.edu).

Deep Learning Enhancement and Privacy-Preserving Deep Learning: A Data-Centric  
Approach

by

Hung S. Nguyen

A dissertation submitted in partial fulfillment  
of the requirements for the degree of  
Doctor of Philosophy  
Department of Electrical Engineering  
College of Engineering  
University of South Florida

Major Professor: Morris J. Chang, Ph.D.  
Ismail Uysal, Ph.D.  
Zhuo Lu, Ph.D.  
Lu Lu, Ph.D.  
Simon Ou, Ph.D.

Date of Approval:  
June 18, 2023

Keywords: Dimension reduction, Non-IID data, Privacy preservation, Federated learning

Copyright © 2023, Hung S. Nguyen

## **Dedication**

This dissertation is dedicated to my dear family, including my parents, my sister, and her husband. They initiated my Ph.D. journey by encouraging me to go study abroad. They have supported me a lot throughout my entire studying life. Without their endless mental and economic support, I could not have reached this step.

I am also dedicating this to my beloved wife and in-law family members, including my parents-in-law, sisters-in-law, and brothers-in-law. They all mentally supported me a lot and kept encouraging me to overcome difficulties. My wife especially had to give up her dream job to come and help me through my Ph.D. journey. Without their support, this work was not possible to complete.

While working on this dissertation, our first daughter arrived, becoming a precious source of immense joy and inspiration in my life. As I embarked on this academic journey, you entered the world, filling it with love and wonder. Your arrival has taught me the true meaning of perseverance and resilience. Amidst sleepless nights and demanding schedules, you have been my guiding light, reminding me of the importance of balance and the power of unconditional love. May this dissertation serve as a testament to my unwavering dedication to both my scholarly pursuits and the cherished role of being your parent. You are my greatest motivation, and I dedicate this achievement to you with boundless love and gratitude.

## **Acknowledgments**

First, I would like to express my sincerest gratitude to my advisor, Professor J. Morris Chang, for his priceless guidance through my Ph.D. journey over the last five years. I want to thank Prof. Chang for unconditionally supporting me with academic knowledge, research instructions, and professional career advice, particularly guiding me to choose research topics that are among the most interesting in the industry and academia.

I would also like to sincerely thank the professors on my dissertation committee, Dr. Ismail Uysal, Dr. Xinming Ou, Dr. Lu Lu, and Dr. Zhuo Lu. The knowledge that I have gained from them during coursework and discussions helped enrich this dissertation.

Last but not least, I would like to express my deep gratitude to my academic friends who have been working around me over the past five years. This dissertation would not have been possible without their help.

## Table of Contents

List of Tables . . . . .	iv
List of Figures . . . . .	v
Abstract . . . . .	vii
Chapter 1: Introduction . . . . .	1
Chapter 2: Federated Learning for Skewed Distribution . . . . .	2
2.1 Introduction . . . . .	2
2.2 Scenario . . . . .	5
2.3 Preliminary: Masked Autoencoder for Distribution Estimation . . . . .	5
2.4 Federated Learning for Skewed Distribution Using Sample Weights . . . . .	6
2.4.1 Sample Weights Design . . . . .	8
2.4.2 Probability Density Approximation . . . . .	10
2.4.3 Sample Weight Approximation . . . . .	11
2.4.4 Learning on Skewed Distribution Data . . . . .	12
2.5 Privacy Leakage Analysis . . . . .	13
2.6 Experiments . . . . .	14
2.6.1 Datasets and Non-IID Setting . . . . .	14
2.6.1.1 Simulated Non-IID: MNIST . . . . .	14
2.6.1.2 Real Non-IID: Femnist Dataset . . . . .	15
2.6.1.3 Real Non-IID: Chest Xray Dataset . . . . .	15
2.6.1.4 Data Examples . . . . .	15
2.6.2 Implementation Detail . . . . .	16
2.6.2.1 Baselines . . . . .	16
2.6.2.2 Federated Learning Classification Model . . . . .	17
2.6.2.3 Density Estimation Model (MADE) . . . . .	17
2.6.2.4 Sample Weight Approximation . . . . .	18
2.6.3 Results . . . . .	18
2.6.3.1 Classification Accuracy . . . . .	19
2.6.3.2 Effective Communication Rounds . . . . .	20
2.6.3.3 Effective Communication Cost . . . . .	23
2.7 Conclusion . . . . .	24
Chapter 3: Synthetic Information Towards Maximum Posterior Ratio for Deep Learning on Class Imbalanced Data . . . . .	26
3.1 Introduction . . . . .	26

3.2	Preliminaries . . . . .	30
3.2.1	Imbalance Ratio (IR) . . . . .	30
3.2.2	Evaluation Metrics . . . . .	30
3.2.3	Entropy-based Active Learning . . . . .	31
3.3	The Problem of Learning From Imbalanced Datasets . . . . .	32
3.4	Related Work . . . . .	33
3.4.1	Sampling-based Approach. . . . .	34
3.4.2	Cost-sensitive Learning Approach . . . . .	35
3.4.3	Ensemble Learning Approach . . . . .	36
3.5	Federated Learning for Skewed Distribution Using Sample Weights . . . . .	36
3.5.1	Methodology Motivation . . . . .	36
3.5.2	Generating Minority Synthetic Data . . . . .	37
3.5.3	Algorithm . . . . .	43
3.6	Algorithm Time Complexity . . . . .	43
3.7	Experiments . . . . .	45
3.7.1	Experimental Setup . . . . .	45
3.7.1.1	SIMPOR Settings . . . . .	47
3.7.1.2	Evaluation Classification Settings . . . . .	47
3.7.2	SIMPOR on Artificial Moon Dataset . . . . .	48
3.7.3	SIMPOR on Forty-one Real Datasets . . . . .	51
3.7.3.1	Classification Results . . . . .	51
3.7.3.2	Statistical Test . . . . .	56
3.7.3.3	Data Visualization . . . . .	57
3.7.3.4	Processing Time . . . . .	59
3.7.4	Empirical Study on the Impact of Radius Factor $r$ . . . . .	59
3.7.5	Empirical Study on the Impact of Informative Portion . . . . .	61
3.8	Conclusion . . . . .	62
Chapter 4: AutoGAN-Based Dimension Reduction for Privacy Preservation . . . . .		64
4.1	Introduction . . . . .	64
4.2	Related Work . . . . .	67
4.2.1	Literature Review . . . . .	67
4.2.2	Preliminaries . . . . .	69
4.2.2.1	Auto-encoder . . . . .	70
4.2.2.2	GAN . . . . .	70
4.3	Methodology . . . . .	71
4.3.1	Problem Statement . . . . .	72
4.3.2	Threat Model . . . . .	72
4.3.3	$\epsilon$ -Dimension Reduction Privacy ( $\epsilon$ -DR Privacy) . . . . .	73
4.3.4	AutoGAN Dimension Reduction for Privacy Preserving . . . . .	74
4.3.5	Optimization with Constraint . . . . .	77
4.3.6	Training Algorithms . . . . .	77
4.4	Experiments and Discussion . . . . .	79
4.4.1	Experiment Setup . . . . .	81
4.4.2	Utility . . . . .	82

4.4.3 Privacy . . . . .	83
4.5 Comparison to GAP[18] . . . . .	83
4.6 Visual Comparison to Privacy Preserving Techniques . . . . .	87
4.7 Conclusion . . . . .	89
References . . . . .	91
Appendix A: Copyright Permissions . . . . .	105

## List of Tables

Table 2.1	Effective Communication Rounds for exchanging model weights. . .	21
Table 2.2	Communication overhead each round per client. . . . .	21
Table 3.1	Dataset description. . . . .	46
Table 3.2	Classification model settings for each dataset. . . . .	48
Table 3.3	Classification result on Moon dataset. . . . .	49
Table 3.4	F1-score over different datasets. . . . .	52
Table 3.5	AUC result over different datasets. . . . .	53
Table 3.6	Precision results over 41 datasets. . . . .	54
Table 3.7	Recall results over 41 datasets. . . . .	55
Table 3.8	Wilcoxon signed rank hypothesis test results. . . . .	57
Table 3.9	Processing time over 41 datasets . . . . .	60
Table 4.1	Implementation information . . . . .	80
Table 4.2	Sample visualization of AutoGAN, DP, PCA over three datasets . .	88



## List of Figures

Figure 2.1	FedDisk framework. . . . .	7
Figure 2.2	Example images from MNIST, FEMNIST and Chest Xray datasets.	16
Figure 2.3	Classification performance over global iterations. . . . .	18
Figure 2.4	Classification statistics over 100 clients. . . . .	19
Figure 2.5	Validation and train losses during training the global MADE. . . . .	20
Figure 2.6	Summary of Effective Communication Cost over 3 datasets. . . . .	22
Figure 2.7	Global model’s loss over communication rounds. . . . .	22
Figure 3.1	Learning from imbalanced datasets . . . . .	33
Figure 3.2	Demonstration of synthetic sample generation. . . . .	41
Figure 3.3	Artificial class imbalanced Moon dataset with IR of 7:1. . . . .	49
Figure 3.4	Data and model decision boundary visualization for Moon Dataset.	50
Figure 3.5	Winning times over 41 datasets. . . . .	56
Figure 3.6	Data visualization over methods. . . . .	58
Figure 3.7	F1-score and AUC results with varying Gaussian standard deviation.	61
Figure 3.8	F1-score and AUC results with varying informative portion IP. . . . .	62
Figure 4.1	Attack model. . . . .	71
Figure 4.2	DR projection and reconstruction. . . . .	73
Figure 4.3	AutoGAN-DRP . . . . .	74
Figure 4.4	Accuracy for different number of reduced dimensions. . . . .	79
Figure 4.5	Average distance measurement result. . . . .	79
Figure 4.6	AutoGAN-DRP and GAP visualization. . . . .	85

Figure 4.7 GENKI Facial Expression Vs AutoGAN-DRP Distance. . . . . 86

## **Abstract**

Deep Learning and its applications have become attractive to a lot of research recently because of its capability to capture important information from large amounts of data. While most of the work focuses on finding the best model parameters, improving machine learning performance from data perspective still needs more attention. In this work, we propose techniques to enhance the robustness of deep learning classification by tackling data issue. Specifically, our data processing proposals aim to alleviate the impacts of class-imbalanced data and non- IID data in deep learning classification and federated learning scenarios. In addition, data pre-processing strategies such that dimensionality reduction is also enhanced using a proposed deep learning-based technique for a scenario of data privacy preservation. By conducting several experiments and comparisons, we show that our approaches yield good performance and constantly outperform many state-of-the-art methods.

## Chapter 1: Introduction

In recent decades, deep learning has become successful in many automatic tasks such as pattern recognition, natural language processing, image and vision computing by extracting and learning from large amounts of data. While most existing works focus on enhancing machine learning models to improve task performance, data quality improvement is still in lacks of attention. There are numerous aspects to look for in order to improve data quality, such as data augmentation, data dimensionality reduction, data imbalance, missing data, and feature extraction. In practice, by improving data quality, we could significantly boost machine learning performance. In this work, we aim to enhance machine learning performance by focusing on data processing techniques to alleviate the negative impacts of class imbalance and non-IID data (not independent and identically distributed). Besides, we also propose a deep learning-based dimensionality reduction technique to improve classification accuracy and preserve user privacy simultaneously. Proposed techniques are introduced in the following chapters. Specifically, Chapter 2 introduces a method to tackle the non-IID issue in federated learning, which might significantly reduce machine learning performance. Chapter 3 introduces a class balancing technique that generates synthetic data for minor classes. This could help deep learning to avoid slow convergence problems and task performance. In Chapter 4, we propose a dimensionality reduction technique based on the state-of-the-art generative models (i.e., AutoEncoder and Generative Adversarial Network) to not only improve classification accuracy but also preserve data privacy.

## Chapter 2: Federated Learning for Skewed Distribution

### 2.1 Introduction

Since the demand for massive data in artificial intelligent machines, the concept of federated learning (FL) was first introduced in 2017 [69], which is a collaboratively decentralized learning framework. In contrast to centralized learning approaches (in which datasets are sent to an aggregator), FL encourages data holders to contribute without the privacy concern of exposing their raw data. For example, several hospitals holding patient records would participate in a machine learning system to provide better disease predictions via a FL framework without the concern of privacy disclosure. Since then, FL has been seen in various applications in different fields [40, 98, 29, 102].

To learn a model utilizing data from multiple clients without directly accessing to clients' data, authors in [69] introduced Federated Averaging (FedAvg) and demonstrated its robustness. The main idea is that clients (data holders) involve in a model training process by exchanging local models' weights instead of exchanging raw data. One of the main concerns in FL is that the data might come from different sources and have different distributions. Thus, FL performance is significantly reduced because this violates a fundamental machine learning assumption that data should be independent and identically distributed (IID). The FL over non-IID data has been shown in existing works [107, 79, 46, 62, 86, 89, 57, 104] that its performance deteriorates dramatically. In this work, we focus on tackling the non-IID data issue in a federated learning system, in which the collected data feature distribution is skewed. The skewness might be caused by many different reasons. For example, clients might perform different sampling methods, apply different normalization methods, or sample using different devices.

Over the past few years, there have been a number of approaches aiming at reducing non-IID data impacts. While many current works focus on the skewed label distribution, there are only limited approaches considering skewed feature distribution data which is very common in various fields, e.g., medical images collected from different x-ray machines. Authors in [105] explained the performance reduction as the problem of weight divergence. They then proposed an alleviation by combining local data with global shared data to train each client. However, it raises the concern of privacy violation with the shared data. Li et al. illustrated in their work (FedBN) [59] that Local Batch Normalization would help to reduce the problem of non-IID data. FedBN suggests clients to not synchronize local batch normalization parameters with the global model. Sahu et al. introduce FedProx [78] to solve the weight-divergence issue by proposing a loss function which constrains the local models to stay close to the global model. FedDNA [24] shares statistical parameters of models (means and variances) and aims at finding averaging weights for each client’s model to minimize models’ weights divergence across clients. However, as this only considers the aggregating weights for each model, the improvement is minor. FedNova [91] suggests to normalize local weights before synchronizing with the aggregator. FedMA [90], AFL [71] and PFNM [101] consider combinations of layer-wise parameters and provide an aggregation of such parameters to alleviate the non-IID issue. In FedRod [16], Chen and Chao deal with the non-IID issue by learning hyper-networks locally which results in personalized classifiers for clients and clients’ class distributions. Recently, Tan et al. [88] tackle the non-IID data issue by exchanging representation vectors of samples in a given class instead of model’s parameters, enable clients to have personalized model architecture. However, these suggestions do not directly consider the data distribution skewness at the data level, which could lead to performance reduction and convergence slowness.

In Federated Learning (FL), when dealing with non-IID data, the primary problem is the divergence of weights, which worsens as the data distribution becomes more skewed, as found in [105] by Zhao et al.. According to Zhao et al.’s research, this issue arises due to differences

between client individual and global distributions. To address this problem, we proposed an algorithm that utilizes sample weights to adjust individual client distributions closer to the global distribution during the training process. However, obtaining global information across clients is challenging in an FL setting because clients do not allow the exposure of their raw data. To overcome this challenge, the proposed method implicitly shares statistical information of client data without revealing the client’s raw data. The method only requires clients to exchange additional model weights using a typical FL procedure. Once the adjustment weights are acquired, the machine learning model can be trained using a standard FL framework. The proposed method is demonstrated to improve FL accuracy and significantly reduce FL communication costs through experiments on three real-world datasets.

Our contributions are as follows:

1. Provide a theoretical base to deal with skewed feature distribution data for federated learning by adjusting sample weights derived from the machine learning empirical risk.
2. Provide a practical solution to mitigate the problem of learning from non-IID data for the FL framework without sharing clients’ draw data. It not only helps to improve the classification accuracy of the global model but also accelerates the model convergence process, thus minimizing communication costs.
3. Several experiments were conducted on three datasets, including MNIST, non-IID benchmark dataset FEMNIST and real-world dataset Chest-Xray. The results demonstrate that the proposed method outperforms other experimental methods in classification accuracy and dramatically reduces the communication cost.
4. As the proposed method needs to exchange additional information, we also provide a theoretical analysis to analyze the potential privacy leakage. We showed that the leakage information becomes insignificant when the number of clients increases.
5. To our best knowledge, the proposed method is the first method utilizing data distribution information and sample weights to tackle the FL Non-IID issue.

The rest of this paper is organized as follows. Section 3.3 introduces our problem in a scenario where clients hold different distribution datasets. Section 2.3 introduces a neural network-based model that is leveraged in our work to carry density information. Our proposed solution is introduced in Section 2.4. We provide a privacy leakage analysis in Section 2.5 as the proposed method indirectly exchanges distribution information. Section 3.7 shows our experimental results and illustrates the proposed method’s performance. Section 3.8 summarizes our study and discusses future work to improve the proposed method.

## 2.2 Scenario

In this section, we introduce and formulate the scenario of FL with skewed feature distribution across clients. Our scenario is a learning collaboration between  $K$  clients to build a global classification model that maximizes the global accuracy given arbitrary data. Each client holds a number of individual records that they are not willing to share with others due to privacy concerns. This study focus on preventing the performance of the global model from deteriorating because of the distribution skewness issue [58] across clients.

We denote the data and associated labels held by client  $k \in \{1, \dots, K\}$  as  $\{(\mathbf{x}_k^i, y_k^i)\}_{i=1}^{N_k}$  where  $\mathbf{x}_k^i \in \mathbb{R}^d$  and  $y_k^i \in \mathbb{N}$ . Instead of learning each model for every client  $f(\mathbf{w}_k)$  (where  $f(\cdot)$  demotes local inference models and  $\mathbf{w}_k$  is the model’s parameter of the  $k^{\text{th}}$  client), the objective is to maximize the performance of a global model  $g(\mathbf{w})$  ( $g(\cdot)$  approximates the global inference model and  $\mathbf{w}$  is the global model’s parameter) that is resilient to data skewness.

## 2.3 Preliminary: Masked Autoencoder for Distribution Estimation

The proposed method asks the clients to share additional model weights that carry their local datasets’ distribution information instead of sharing the raw data. We utilize a neural network-based density estimation, namely, Masked Autoencoder for Distribution Estimation (MADE) [34]. This section briefly introduces MADE.



MADE is designed to estimate the probability distribution of input components (e.g., pixels in an image). MADE assumes input components are dependent instead of independent, which is relevant in many applications. For example, MADE can decompose the distribution of an instance  $\mathbf{x}$  consisting  $n$  components  $x_1, x_2, x_3, \dots, x_n$  as follows:

$$p(\mathbf{x}) = p(x_1|x_2, x_3, \dots, x_n) \cdot p(x_2|x_3, \dots, x_n) \dots p(x_{n-1}|x_n) \cdot p(x_n). \quad (2.1)$$

In our study, the instances are images and each pixel can be considered as a component. Thus,  $n$  is the size of a flatten image vector.

For MADE implementation, a shallow neural network is utilized. Its input and output size are equal (similar to an Autoencoder), for example a size of  $n$  for the above example. The main idea is to mimic Equation 2.1 by masking neuron connections across layers to control the seen and unseen connections to model output. Specifically, MADE poses constraints on the model that each output component in a certain layer only connects to its dependent input components in the previous layer. Masks are created based on such principle, and applied to the weights of the model.

Specifically, MADE assigns each unit in a hidden layer an integer  $m$  between 1 and  $D - 1$ , where  $D$  is the number of dimensions. Denote  $m(k)$  as the maximum number of units in the previous layer to which the  $k^{\text{th}}$  hidden unit can connect, the weight mask  $M$  is then formulated as follows:  $M_{k,d} = 1_{m(k) \geq d} = \begin{cases} 1 & \text{if } m(k) \geq d \\ 0 & \text{otherwise,} \end{cases}$  for  $d \in \{1, \dots, D\}$  and  $k \in \{1, \dots, K\}$  with  $K$  being the number of hidden layer units.

## 2.4 Federated Learning for Skewed Distribution Using Sample Weights

In this section, we propose a solution to alleviate the negative impact of distribution skewness across clients for federated learning. The proposed method aims to find weights for training samples in order to adjust the global distribution. To achieve this goal, we need

to exchange some statistical information between clients and the aggregator in a privacy-preserving manner. The remainder of this section introduces how we design sample weights, how we exchange statistical information without exposing clients’ raw data, and how we derive sample weights from achieved information. After achieving weights for the samples, the training process for machine learning tasks is similar to FedAvg [69]. Our framework is illustrated in Figure 2.1. The proposed method, namely FedDisk, requires a 2-phase process. First, clients jointly learn a global density estimation model and their local density models utilizing MADE models. These models are then used to derive sample weights for the local training process. Second, the machine learning tasks can be learned by the conventional FL procedure, with the data skewness issue mitigated by the sample weights from the first phase.

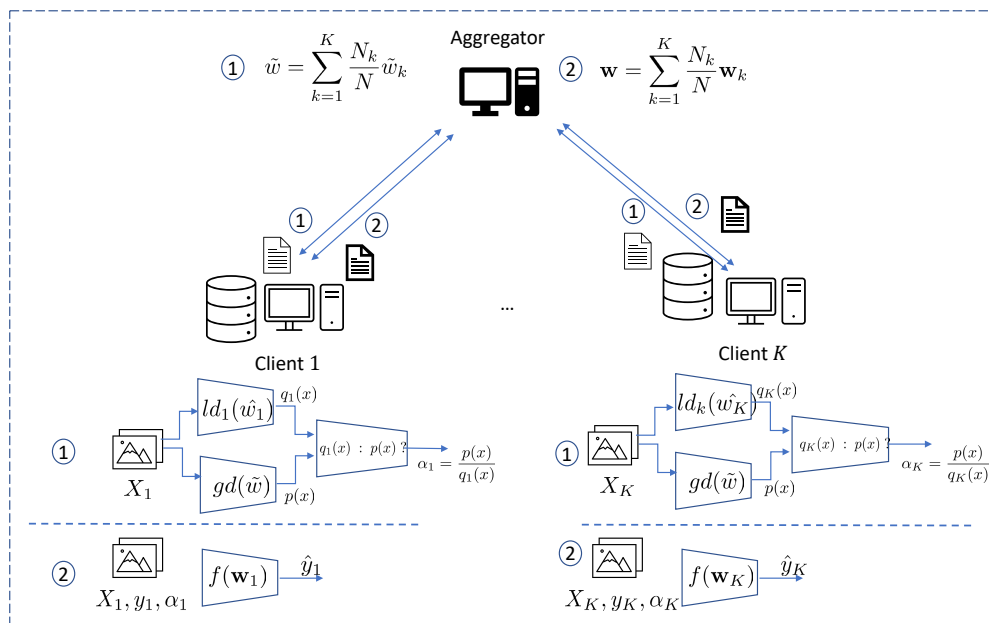


Figure 2.1: FedDisk framework. The proposed framework has two phases. First, local and global probability density functions ( $p(x), q(x)$ ) are estimated via MADE models leveraging FL procedures. Then, the sample weights  $\alpha$  are computed by approximating density ratio via class probability estimation. Second, the machine learning tasks (e.g., classification) can be performed similar to a typical FL method (i.e., FedAvg) with the sample weights acquired from phase 1.

### 2.4.1 Sample Weights Design

As we do not have sufficient information about the true distribution, we consider the combination of all clients' dataset distribution as our true distribution. Thus, we consider the probability density function (pdf) of the true distribution as

$$p(\mathbf{x}) = \sum_{k=1}^K q_k(\mathbf{x}), \quad (2.2)$$

where  $q_k(\mathbf{x})$  represents the pdf of the  $k^{\text{th}}$  client's data.

To jointly learn a global model, the system finds the expectation of the loss function  $l(g(\mathbf{x}), y)$  with sample  $\mathbf{x}$  that drawn from the true distribution. The expected loss is formulated by the associated risk [49] as follows:

$$\mathbb{E}[l(g(\mathbf{x}), y)] = \iint l(g(\mathbf{x}), y) p(y|\mathbf{x}) p(\mathbf{x}) d\mathbf{x} dy, \quad (2.3)$$

where  $p(\mathbf{x}, y)$  is the joint pdf of a sample  $\mathbf{x}$  and its associated label  $y$ , and  $p(y|\mathbf{x})$  is the conditional probability of a label  $y$  given a sample  $\mathbf{x}$ . We also assume that for any client  $k \in \{1, \dots, K\}$  with local data distribution  $q_k(\mathbf{x})$ , the conditional probability of a label  $y$  given a sample  $\mathbf{x}$  is equivalent to that of the true distribution, namely

$$q_k(y|\mathbf{x}) = p(y|\mathbf{x}). \quad (2.4)$$

From Equation 2.2, 2.3, 2.4, and by multiplying with factor  $\frac{q_k(\mathbf{x})}{q_k(\mathbf{x})} = 1$ , the expected loss in Equation 2.3 can be expanded as follows:

$$\mathbb{E}[l(g(\mathbf{x}), y)] = \iint l(g(\mathbf{x}), y)p(y|\mathbf{x})p(\mathbf{x})d\mathbf{x}dy, \quad (2.5)$$

$$= \iint l(g(\mathbf{x}), y)q_k(y|\mathbf{x})\frac{q_k(\mathbf{x})}{q_k(\mathbf{x})}p(\mathbf{x})d\mathbf{x}dy \quad (2.6)$$

$$= \iint l(g(\mathbf{x}), y)q_k(\mathbf{x}, y)\frac{p(\mathbf{x})}{q_k(\mathbf{x})}d\mathbf{x}dy. \quad (2.7)$$

The objective of the global model thus amounts to minimize the empirical risk over all  $K$  clients' datasets:

$$\underset{g}{\text{minimize}} \sum_{k=1}^K \frac{1}{N_k} \sum_{j=1}^{N_k} \alpha_k^j l(g(\mathbf{x}_k^j), y_k^j), \quad (2.8)$$

where  $\mathbf{x}_k^j, y_k^j$  are the  $j^{\text{th}}$  sample and its label.  $N_k$  is the number of samples in the  $k^{\text{th}}$  client's dataset.  $\alpha_k^j$  is the corresponding sample weight computed as

$$\alpha_k^j = \frac{p(\mathbf{x})}{q_k(\mathbf{x})} = \frac{\sum_{i=1}^K q_i(\mathbf{x})}{q_k(\mathbf{x})}. \quad (2.9)$$

Our problem becomes minimizing the summation of the loss functions (Equation 2.8) over all clients. For each client, the loss function is minimized over local samples with the corresponding  $j^{\text{th}}$  sample weight of the client  $k^{\text{th}}$ ,  $\alpha_k^j$ . The sample weights could be estimated by the density ratio between the true distribution (global distribution) and the client distributions (local distributions). For each client, the local distribution can be estimated using its local data. However, the challenge is to achieve the true distribution without having access to other clients' data. To solve this, we leverage a neural network-based density estimation model to learn the global density function via a typical federated learning procedure. Thus, clients can implicitly exchange some statistical information, while still preserving the privacy in client data.

### 2.4.2 Probability Density Approximation

To estimate global density and preserve client privacy at the same time, we propose to leverage a neural network-based density estimation so that we can exchange local density information (via models' weights) with the aggregator without sharing the raw data. In this work, we leverage a well-known method, namely, Masked Autoencoder for Distribution Estimation (MADE, [34]), which is briefly reviewed in Section 2.3. Elaborately, each client aims to estimate its local probability density  $q_k(\mathbf{x})$  using its own dataset, and all  $K$  clients jointly estimate the global probability density  $p(\mathbf{x}) = \frac{N_1}{N}q_1(\mathbf{x}) + \dots + \frac{N_k}{N}q_k(\mathbf{x})$ . Learned MADE models are used to approximate local probability density functions, and the global MADE model approximates the global probability density. The learning process is described as follows.

The  $k^{th}$  client learns a local density estimation model  $ld_k(\hat{w}_k)$  (where  $ld(\cdot)$  approximates density estimation function with parameter  $\hat{w}$ ) using its local data. It then jointly learns a global density estimation model  $gd(\tilde{w})$  (where  $gd(\cdot)$  represents the global density function with the parameter  $\tilde{w}$ ) using the procedure similarly to FedAvg [69]. Specifically, for the local model density estimation models, each client train a MADE model on its local data until the loss function can not be improved. For the global density estimation model, each client trains its data locally for a certain number of iterations, and then model parameters are sent to an aggregator for the aggregation. Since clients might own different number of samples, a weight of  $N_k/N$  (where  $N_k$  and  $N$  are the number of samples of the  $k^{th}$  client and the the total number of samples over all clients) is used for adjusting client parameter significance, similar to FedAvg. After aggregating all clients' model parameters, the aggregator shares global model parameters to all clients. The steps are repeated until the validation loss starts increasing. The global MADE model aggregation from  $K$  clients at iteration  $t$  can be described as follows:

$$\tilde{w}^t = \sum_{k=1}^K \frac{N_k}{N} \tilde{w}_k^t \quad (2.10)$$

### 2.4.3 Sample Weight Approximation

After the local and global density approximations by MADE models are fully learned (Section 2.4.2), we can estimate sample weights in Equation 2.9. Since MADE models output vectors of conditional probabilities for each element in the  $d$ -dimensional input  $\mathbf{x}$ , an intuitive way to compute  $p(\mathbf{x})$  is to multiply all the conditional probabilities. However, as  $p(\mathbf{x})$  vanishes when any of the conditional probabilities vanishes, we instead keep the output as a vector of conditional probabilities (same size as input) and approximate the density ratio in Equation 2.9 using a class probability estimation method inspired by [70]. The method aims at training a binary classifier to output a probability that represents the ratio between  $p(\mathbf{x})$  and  $q(\mathbf{x})$ . The solution detail is described in the rest of this subsection.

After each client receives the final global MADE model and trains its own local MADE, it starts to evaluate sample weights for its local data. The training data of the  $k^{\text{th}}$  client,  $X_k$ , is then fed into both the global MADE (the global MADE is downloaded to clients so that this step can be done locally) and the local MADE to estimate  $p(\mathbf{x})$  and  $q_k(\mathbf{x})$ , respectively. Denote  $\mathbf{u}$  as the output vector of density estimation models, and  $l$  be the pseudo label indicating whether it is sampled from the global destination ( $l = 1$ ) or the local distribution ( $l = 0$ ). Each client then trains a binary classifier to differentiate whether the output  $\mathbf{u}$  comes from  $p(\mathbf{x})$  or  $q_k(\mathbf{x})$ . Outputs of the two MADE models (the sample size of each output is  $N_k$ ) are concatenated to a new vector dataset including samples  $\{(\mathbf{u}_k^i, l_k^i)\}_{i=1}^{2N_k}$ , and is used to train the binary classifier. The conditional probabilities of the binary classification model  $h(\mathbf{u}, w_h)$  (where  $\mathbf{u}$  is the input variable,  $w_h$  is the model parameter) can be approximated as following:

$$\mathcal{P}(\mathbf{u}|l = 0) \propto q_k(\mathbf{x}), \quad \mathcal{P}(\mathbf{u}|l = 1) \propto p(\mathbf{x}). \quad (2.11)$$

From Bayes' rule, we have

$$\frac{p(\mathbf{x})}{q(\mathbf{x})} = \frac{\mathcal{P}(\mathbf{u}|I=1)}{\mathcal{P}(\mathbf{u}|I=0)} = \left( \frac{\mathcal{P}(I=1|\mathbf{u})\mathcal{P}(\mathbf{u})}{\mathcal{P}(I=1)} \right) \left( \frac{\mathcal{P}(I=0)}{\mathcal{P}(I=0|\mathbf{u})\mathcal{P}(\mathbf{u})} \right) \quad (2.12)$$

$$= \frac{\mathcal{P}(I=1|\mathbf{u})\mathcal{P}(I=0)}{\mathcal{P}(I=0|\mathbf{u})\mathcal{P}(I=1)}. \quad (2.13)$$

We approximate the marginal probability ratio between two distributions ( $\mathcal{P}(I=0)$  and  $\mathcal{P}(I=1)$ ) by the number of samples from the two distributions  $N_k$  over the concatenated dataset size ( $2N_k$ ). Thus, We have  $\frac{\mathcal{P}(I=0)}{\mathcal{P}(I=1)} = \frac{N_k}{2N_k} \frac{2N_k}{N_k} = 1$ .

The density ratio then can be estimated as follows:

$$\frac{p(\mathbf{x})}{q(\mathbf{x})} = \frac{\mathcal{P}(I=1|\mathbf{u})}{\mathcal{P}(I=0|\mathbf{u})} = \frac{\mathcal{P}(I=1|\mathbf{u})}{1 - \mathcal{P}(I=1|\mathbf{u})}. \quad (2.14)$$

, where  $\mathcal{P}(I=1|\mathbf{u})$  is the classifier's probability-liked output indicating how likely an input vector  $\mathbf{u}$  comes from the global probability  $p(\mathbf{x})$ .

To summarize, the  $j^{\text{th}}$  training sample of client  $k$ ,  $\mathbf{x}_k^j$ , is fed into the client's local MADE model to achieve its corresponding density estimation  $\mathbf{u}_k^j$ .  $\mathbf{u}_k^j$  is then fed into the binary classification function  $h(\mathbf{u})$  to achieve the class probability  $\mathcal{P}(I=1|\mathbf{u}_k^j)$ . This is used to estimate the sample weight  $\alpha_k^j$  (Equation 2.9) based on Equation 2.14. In words, the binary classification model  $h(\mathbf{u}, \mathbf{w}_h)$  is expected to return higher weights for samples that are likely belonging to the true distribution and vice versa.

#### 2.4.4 Learning on Skewed Distribution Data

After acquiring sample weights, each client starts to train the model on the local dataset and corresponding sample weights for a machine learning task (e.g., classification) as a typical FL framework. In this work, we follow the procedure introduced by FedAvg to learn the

global model. The aggregator aggregates clients' local models as follows:

$$\mathbf{w}^t = \sum_{k=1}^K \frac{N_k}{N} \mathbf{w}_k^t \quad (2.15)$$

where  $\mathbf{w}^t$  and  $\mathbf{w}_k^t$  are the global and local model parameter of  $k^{\text{th}}$  client at the  $t^{\text{th}}$  iteration.

## 2.5 Privacy Leakage Analysis

In this section, we discuss the privacy leakage of our method compared to the conventional FL. Similar to many other works, we utilize additional information to alleviate the negative impact of non-IID data, i.e., parameters of MADE models. However, these parameters might contain distribution information of clients' data. However, we prove that the more clients are involved in the FL training process, the less our extra information is leaked. Although clients might have access to the global distribution via the global MADE model, a client can't infer other individual local distribution when the number of clients is large. The detail is described as follows.

Assume each client samples their own data point  $\hat{Z}_k \sim Q_k$  independently, and let  $\Theta$  be a random variable taking values in  $\llbracket 1, K \rrbracket$  with  $\mathcal{P}[\Theta = k] = \kappa_k$  and independent of  $\hat{Z}_k$  for each  $k \in \llbracket 1, K \rrbracket$ . Note that  $\hat{Z}_\Theta \sim P$ , and one may quantify the privacy leakage of client  $k$ 's data through the knowledge of  $P$  by the mutual information between  $\hat{Z}_k$  and  $\hat{Z}_\Theta$ , as given by

$$\begin{aligned} I(\hat{Z}_k; \hat{Z}_\Theta) &\leq I(\hat{Z}_k; \hat{Z}_\Theta, \Theta) = I(\hat{Z}_k; \Theta) + I(\hat{Z}_k; \hat{Z}_\Theta | \Theta) \\ &= I(\hat{Z}_k; \hat{Z}_\Theta | \Theta) = \sum_{i=1}^K \mathcal{P}[\Theta = i] I(\hat{Z}_k; \hat{Z}_i) \\ &= \kappa_k H(\hat{Z}_k). \end{aligned} \quad (2.16)$$

In other words, the privacy leakage is proportional to  $\kappa_k$ , which decreases to 0 as long as  $\kappa_k = O(1/K)$  and  $K \rightarrow \infty$ .



## 2.6 Experiments

In this section, we conduct several experiments to evaluate the proposed method on non-IID FL scenarios with three real image datasets (MNIST, Chest-Xray and FEMNIST). Our FL system goal is to learn a global classifier leveraging data from all clients. The classification accuracy is used as a metric to evaluate the performance of the proposed method. The communication cost is evaluated by counting the number of iterations needed for clients to exchange model parameters with the aggregator. We compare our method with other state-of-the-art methods, e.i., FedAvg, FedProx, FedBN, and FedROD.

### 2.6.1 Datasets and Non-IID Setting

In this Section, we describe how datasets are used in our experiments. We categorize our dataset into two groups, simulated non-IID dataset (MNIST) and real non-IID datasets (Femnist & Chest-Xray). The first one contains images that have already been combined together so that our partitioning process is considered for sampling from the same contribution. Thus, we added different levels of noise to each client to simulate the feature skewness as inspired by settings in [57], and [88]. The second group’s data are collected from different sources so that they are considered to be non-IID by nature. All the data are normalized and clipped to the range of [0,1] before training. Each client’s data is split to 85% and 15% for training and testing sets, respectively. The detail of the datasets is described as follows.

#### 2.6.1.1 Simulated Non-IID: MNIST

MNIST dataset [23] contains 60,000 (1x28x28) gray scale images of 10 digits (0-9). The number of unique output labels is 10 representing 10 digits. To mimic feature skewness, we split data equally into 100 partitions and add different level of noise to each client’s data as inspired by the skewness simulation in [57]. The noise is drawn from Gaussian distribution with a mean of 0 and different values of standard deviations. More specifically, the  $k^{th}$  client ( $k \in [0, 99]$ ) is added noise with the variance of  $k * x / 100$  where  $x$  is the added noise variance.

### 2.6.1.2 Real Non-IID: Femnist Dataset

FEMNIST dataset is downloaded from <https://leaf.cmu.edu/>, which is considered a benchmark dataset for real non-IID data. It contains handwritten images of 62 digits and characters (corresponding to 62 unique labels) from different writers and strokes. In this study, we randomly select 100 different writers (each of them owns more than 300 images to avoid overfitting) and assign their data to 100 clients. The average sample size of clients is 387.47, and the standard deviation is 83.04. All images are resized to a (32x32) grayscale and normalized to the range of [0,1] before inputting to models.

### 2.6.1.3 Real Non-IID: Chest Xray Dataset

The Chest-Xray dataset, which contains pneumonia and normal chest Chest-Xray images, are collected from different sources (i.e., COVID-19 [19], Shenzhen Hospital [47], and University of California San Diego (UCSD) [51]) with different image sizes, colors and potentially taken from different medical devices. Thus, we consider this dataset non-IID by nature. After partitioning the data into 100 clients, the mean and standard deviation of the client sample size are 325.50 and 63.74, respectively. All images are converted to grayscale and resized to (32x32). There are two unique output labels (binary classification) to predict chest Chest-Xray images are normal or abnormal.

### 2.6.1.4 Data Examples

Figure 2.2 provides several sample images from the three datasets. The MNIST dataset images have various degrees of noise. Besides, the FEMNIST dataset includes images with different writing styles from various sources. The Chest-Xray dataset comprises images with varying resolutions and light conditions, etc. This makes them non-IID across clients.

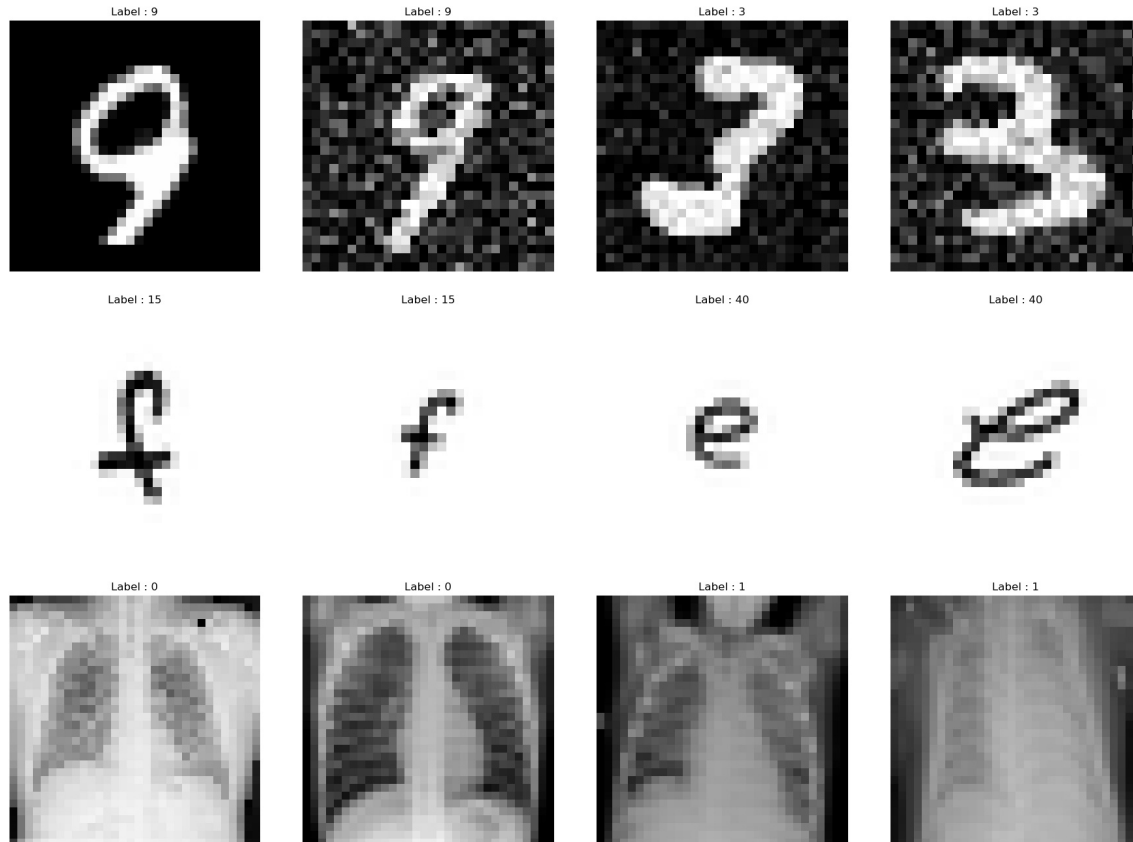


Figure 2.2: Example images from MNIST, FEMNIST and Chest Xray datasets. They are collected from different sources and carried a veraiety of resolutions, styles or conditions.

## 2.6.2 Implementation Detail

### 2.6.2.1 Baselines

We compare our methods with different methods, i.e., FedAvg, FedProx, FedBN, and FedROD. While most implementation details are taken from the initial parameter sets in original papers, we also tune suggested parameters and report the results that give the best values. For FedROD, the results are reported for the hyper-parameter  $\mu$  of 1. We also tried other values in the set 1, 5, 10, 20 and found that the results are very similar. For FedProx, we tuned the parameter  $\mu$  with the candidates of 0.001, 0.01, 0.1, 1 and reported the value of 0.01, 1, 0.01 for the three datasets, Chest-Xray, FEMNIST, MNIST, respectively.

### 2.6.2.2 Federated Learning Classification Model

We use shallow Convolutional Neural Networks (CNN) for the classification of image datasets. The models are constructed by two 5x5 convolutional layers (32 and 32 channels for Chest-Xray, 128 and 128 channels for FEMNIST, 16 and 16 channels for MNIST). Each convolutional layer is followed by 2x2 max pooling and batch normalization layers. A fully connected layer with 16 neurons is added on the top of the models. The input and output sizes are designed to fit each dataset scenario (i.e., image size and the number of unique labels). We use stochastic gradient descent (SGD) with a learning rate of 0.01 for the optimizers. Local iterations are set to 2 for all datasets. Global iterations are set to 1500 for FEMNIST and MNIST, and 500 for Chest-Xray.

### 2.6.2.3 Density Estimation Model (MADE)

Density estimation models (MADE) are constructed by neural networks and the hyper-parameters are taken directly from the initial setting in the original work [34]. Several experiments were conducted to select the optimal set of parameters which yield lowest loss value. The networks include input, output and one hidden layer. The number of neurons in the hidden layer is tuned from a value set of {30, 50, 100, 200, 300, 400}. The final selected number of neurons in the hidden layer are 50, 400, 30 for XRAY, FEMNIST and MNIST datasets, respectively. We noticed that using more neurons than numbers above did not significantly decrease the validation loss, thus they are the optimal settings. The model's input and output sizes are set to the flattened size of images. Specifically, the input and output size for MNIST and FEMNIST datasets is 784 with image sizes of 28x28 pixels. This number is 1024 (32x32 pixels) for Chest-Xray dataset. The maximum training iteration is set to 500, and the training process is stopped when the validation loss starts increasing. Other hyper-parameters are taken directly from [34].

### 2.6.2.4 Sample Weight Approximation

In order to compute the sample weight  $\alpha$ , we use a shallow, fully connected neural network to discriminate the density estimation output vectors coming from which of the two distribution density functions  $p(\mathbf{x})$  or  $q(\mathbf{x})$ . The model contains a 100-neuron hidden layer with Relu activation function. The output layer contains one neuron with Sigmoid activation function. All models applied a learning rate of 0.01, and SGD optimization were used in the training process. The training process is terminated if the loss function is not significantly reduced.

### 2.6.3 Results

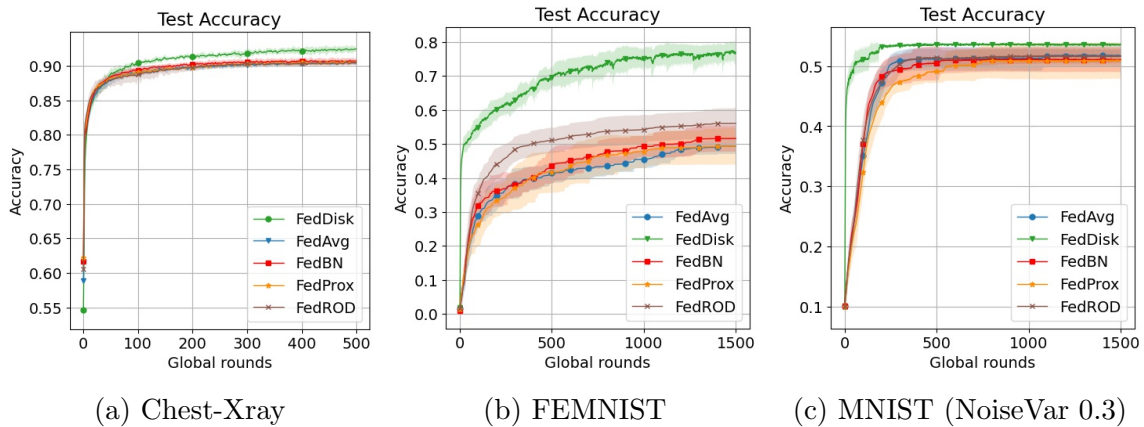


Figure 2.3: Classification performance over global iterations. Global model’s average test accuracy during aggregation process. For MNIST dataset, clients’ data were added noise with the mean of zero and variance of 0.3.

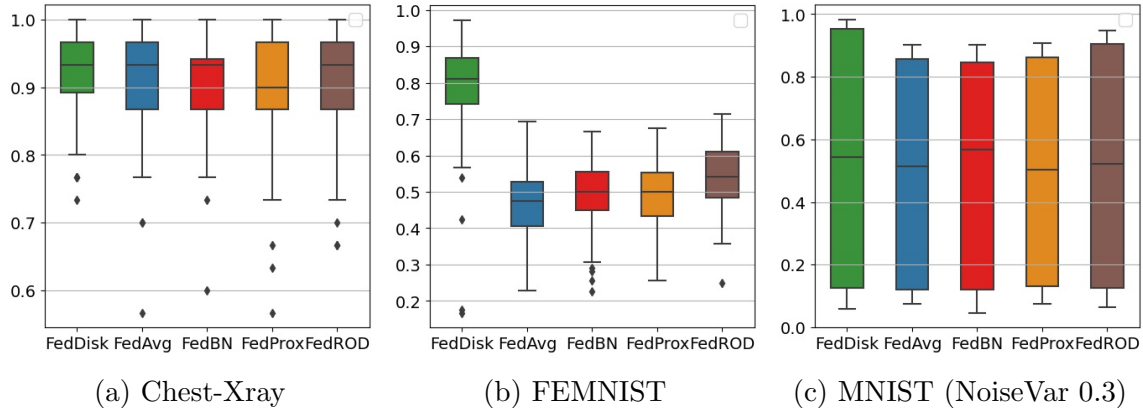


Figure 2.4: Classification statistics over 100 clients. Test accuracy percentiles, min, max and median.

### 2.6.3.1 Classification Accuracy

Figure 2.3 shows the average of the 100 clients’ testing accuracies over training iterations. The shaded regions illustrate the standard deviation over five trials. Overall, FedDisk significantly outperforms other methods in terms of classification accuracy. For example, in Figure 2.3a for Chest-Xray dataset, FedDisk with an accuracy of 92% outperforms others with the highest accuracy of 90.5%. For FEMNIST dataset (Figure 2.3b), our method achieved an accuracy of 78% while others only reached the maximum accuracy of 56% (FedROD). For MNIST, FedDisk reached the accuracy of 54.5% while others only obtained the highest accuracy of 51.7%.

Figure 2.4 shows the descriptive statistical accuracy results of 100 clients on different datasets. The colored rectangles contain 50% of client accuracies. The colored rectangular’s lower and upper edges show the middle values in the first and second half of the sorted clients’ accuracies (lower quartile and higher quartile). The middle dash is the median value. The upper and lower dashes represent the min and max clients’ accuracies. Dots illustrate outliers. Overall, the bars for FedDisk are higher than others, meaning that most clients archive higher accuracy. Dots are also higher (Figure 2.4a and 2.4b) for FedDisk, showing

that outlier clients are also improved. Especially, the bar for FEMNIST is significantly raised for FedDisk, indicating that the proposed method significantly improved for this dataset. It is clear that the proposed method outperforms compared methods in all experimental datasets, including real-world non-IID and simulated non-IID settings.

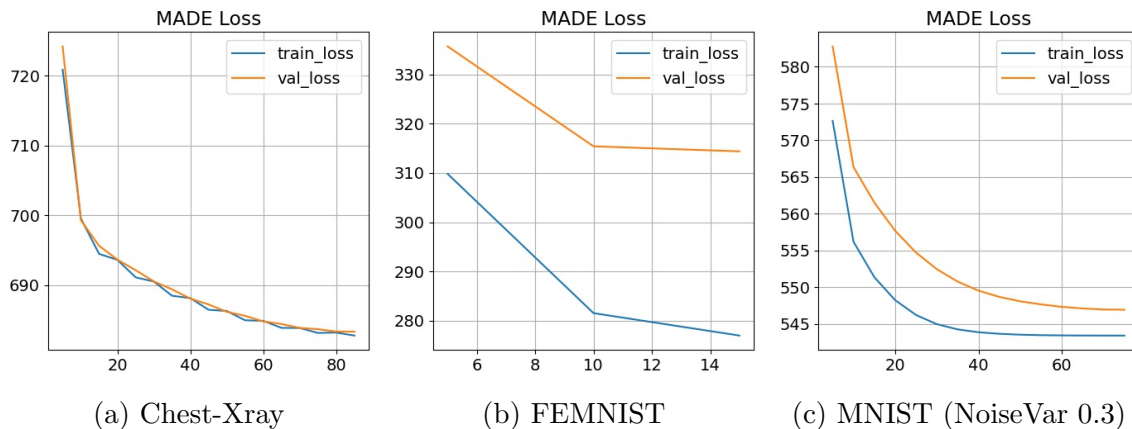


Figure 2.5: Validation and train losses during training the global MADE. The training processes were stopped if the validation loss starts increasing.

### 2.6.3.2 Effective Communication Rounds

To have a fair comparison, we use “Effective Communication Rounds” (ECR) to evaluate effective number of communication iterations for each method. On FedDisk, ECR includes the communication rounds for exchanging MADE models and classification models. Figure 2.5 show the aggregated training loss and validation loss for the global MADE model over communication rounds. The global MADE exchanging process stops when the validation loss starts increasing. For example, the proposed method needs 15 rounds for exchanging global MADE models in the case of the FEMNIST dataset (Figure 2.5b). The ECR for FedDisk in the classification phase is calculated with the number of iterations the method needs to

achieve the highest value among other methods gained. Take the FEMNIST dataset experiment shown in Figure 2.4b for example, FedDisk only needs 105 rounds to reach FedROD’s accuracy at 57% which is the highest accuracy among other experimental methods. Plus 15 rounds to exchange MADE model, FedDisk only needs a total of 120 rounds to effectively reach the top comparison method accuracy. Since other methods don’t need to exchange extra models, the ECRs are calculated by the number of rounds to exchange classification model until they reach their highest accuracy values.

Table 2.1: Effective Communication Rounds for exchanging model weights.

Model	Chest-Xray	FEMNIST	MNIST
FedDisk MADE	80	15	70
FedDisk Classifier	100	105	85
FedDisk Total	180	120	155
FedAvg Classifier	450	1100	500
FedBN Classifier	457	1255	600
FedProx Classifier	440	1200	800
FedROD Classifier	480	1015	550

Table 2.2: Communication overhead each round per client.

Model	Chest-Xray	FEMNIST	MNIST
FedDisk MADE	102000	614000	47000
FedDisk Classifier	39000	447000	12000
FedAvg Classifier	39000	447000	12000
FedBN Classifier	39000	447000	12000
FedProx Classifier	39000	447000	12000
FedROD Classifier	39000	447000	12000



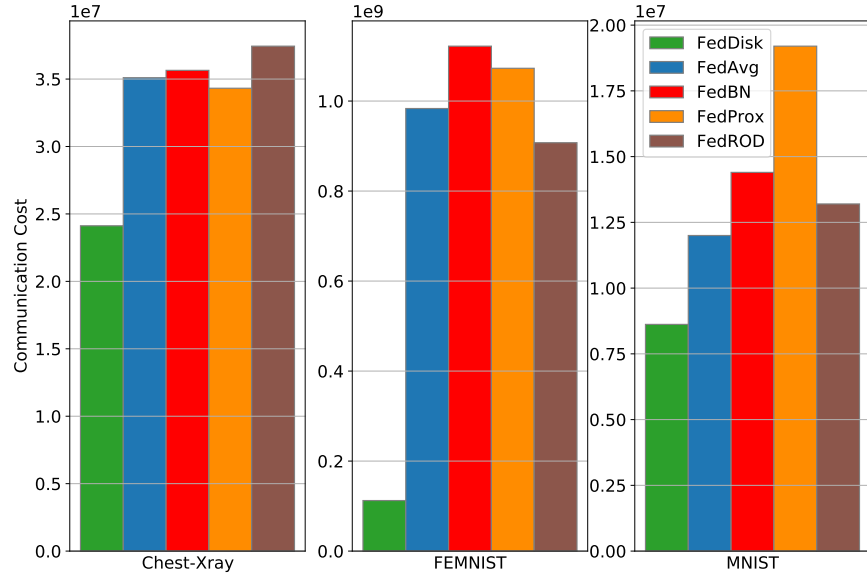


Figure 2.6: Summary of Effective Communication Cost over 3 datasets. The Figure shows that FedDisk is much more efficient in number of communication cost.

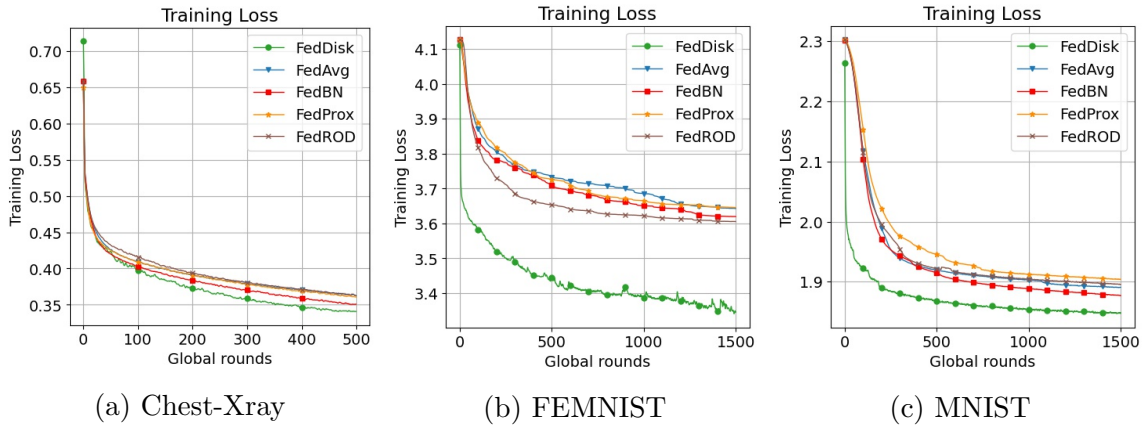


Figure 2.7: Global model's loss over communication rounds. FedDisk loss reduced much faster than other methods.

Table 2.1 shows the summary of “Effective Communication Rounds” for the three experimental datasets. FedDisk mechanism has two phases; one is to transfer MADE models, and the other is to exchange classifiers. The overall FedDisk ECR comprises the communication rounds in the two phases. As shown in the Table, FedDisk is the most effective method as it needs many fewer communication rounds to reach the highest accuracy among other methods. This is because the proposed method only needs a few number communication rounds for the global MADE model to be converged. Besides, the weight-based adjustment converges the global classification model much faster than others. For example, the FedDisk ECR for FEMNIST is only 120 (15 for MADE model exchange plus 105 for classification model exchange), whereas others take more than 1000 iterations.

### 2.6.3.3 *Effective Communication Cost*

To have a comprehensive comparison, the communication cost is estimated for each method. The cost is comprised of communication rounds and overhead, where the overhead is measured by the size of transferred data each round between a client and the aggregator. Since the transferred data mainly contains model weights, our study uses the number of model weights to estimate the overhead size. Table 2.2 shows the number of model weights in different scenarios. Note that a client must consume two costs each communication round; one is for transferring its current model weight, and another is for receiving the updated weight from the aggregator. Generally, the ”effective communication cost” is then computed as follows.

$$C = 2 * S_{cls} * ECR_{cls} \quad (2.17)$$

, where  $C$  is the overall communication cost for one client,  $S_{cls}$  is the overhead (number of transferred classification model weights), and  $ECR_{cls}$  is the number of effective communication rounds. For FedDisk, as it needs to exchange extra MADE in the first phase, the computation

is adjusted as follows.

$$C_{FedDisk} = 2 * (S_{MADE} * ECR_{MADE} + S_{cls} * ECR_{cls}), \quad (2.18)$$

where  $S_{MADE}$  and  $S_{cls}$  are model sizes needed to be transferred in phase 1 (MADE models) and phase 2 (classification models), respectively.  $ECR_{MADE}$  and  $ECR_{cls}$  are the corresponding effective communication rounds in the two phases. Note that for FedDisk,  $ECR_{MADE} + ECR_{cls} = ECR$ , and  $ECR_{cls} = ECR$  for other methods.

Figure 2.6 summarizes effective communication cost. Noticeably, the communication cost reduced significantly for the FEMNIST dataset under the proposed method, eight times, from the second lowest cost method of 907,000,000 (FedROD, brown column) to 112,000,000 (FedDisk, green column). The communication cost reduction trend is also applied to other experimental datasets, Chest-Xray (1.4 times) and MNIST (1.6 times). Thus, the proposed method improves accuracy and dramatically reduces communication costs, one of the most critical concerns in Federated Learning. This is because the loss function were reduced faster as we adjusted using the sample weights. Figure 2.7 demonstrates the global loss values during training. For FedDisk, the sample weights effectively affect the optimization function, and the loss reduces much faster, and the accuracies proportionally increase faster.

## 2.7 Conclusion

In this work, we have proposed a FL method to tackle the issue of distribution skewed data. The technique utilizes a FL framework and a neural network-based density estimation model to derive training sample weights. This helps to adjust the global distribution without revealing clients' raw data. We also provide a privacy analysis for the extra information used in FedDisk (i.e., the parameters of MADE models) and prove that the leakage information becomes less important when the number of clients increases. The experimental results show

that the proposed method not only improves the FL accuracy but also significantly reduces communication costs.

## Chapter 3: Synthetic Information Towards Maximum Posterior Ratio for Deep Learning on Class Imbalanced Data

### 3.1 Introduction

Class imbalance is a common phenomenon; it could be caused by the data collecting procedure or simply the nature of the data. For example, it is difficult to sample some rare diseases in the medical field, so collected data for these are usually significantly less than that for other diseases. This leads to the problem of class imbalance in machine learning. The chance of rare samples appearing in model training process is much smaller than that of common samples. Thus, machine learning models tend to be dominated by the majority class; this results in a higher prediction error rate. Existing work also observed that class imbalanced data cause a slow convergence in the training process because of the domination of gradient vectors coming from the majority class [97, 65].

In the last decades, a number of techniques have been proposed to soften the negative effects of class imbalance for conventional machine learning algorithms by analytically studying particular algorithms and developing corresponding strategies. However, the problem for heuristic algorithms such as deep learning is often more difficult to tackle. As suggested in the most recent deep learning with class imbalance survey [50], most of the works are emphasizing image data, and studies for other data types are missing. Thus, in this work, we focus on addressing the issue of tabular data with class imbalance for deep learning models. We propose a class balancing solution that utilizes entropy-based sampling and data statistical information. As suggested in the survey ([50]) that techniques for traditional ML can be extended to deep learning and inspiring by the comparison in a recent relevant work, Gaussian Distribution Based Oversampling (GDO) [96], we compare the proposed technique

with other widely-used and recent techniques such as GDO [96], SMOTE [14], ADASYN [41], Borderline SMOTE [39], Random Oversampling (ROS).

We categorize existing solutions into model-centric and data-centric approaches in which the first approach aims at modifying machine algorithms, and the latter looks for data balancing techniques. Perhaps data-centric techniques are more commonly used because they do not tie to any specific model. In this category, a simple data balancing technique is to duplicate minority instances to balance the sample quantity between classes, namely random oversampling (ROS). This can preserve the best data structure and reduce the negative impact of data imbalance to some degree. However, this puts too much weight on a very few minority samples; as a result, it causes over-fitting problems in deep learning when the imbalance ratio becomes higher.

Another widely-used technique in this category is Synthetic Minority Oversampling Technique (SMOTE) [14], which randomly generates synthetic data on the connections (in Euclidean space) between minority samples. However, this easily breaks data topology, especially in high-dimensional space, because it can accidentally connect instances that are not supposed to be connected. In addition, if there are minority samples located in the majority class, the technique will generate sample lines across the decision boundary, which leads to distorted decision boundaries and misclassification. To improve SMOTE, Hui Han, *et al.* [39] proposed a SMOTE-based technique (Borderline SMOTE), in which they only apply SMOTE on the near-boundary samples determined by the labels of their neighbors. For example, if a sample Euclidean space-based group includes samples from other classes, they can be considered samples near the border. Since this technique is entirely based on Euclidean distance from determining neighbors to generating synthetic data, it performs poorly in high dimensional space. Similar to SMOTE, if there is any poorly generated sample near the boundary, it will worsen the problem due to synthetic samples bridges across the border. Leveraging the same way as SMOTE generates synthetic samples, another widely-used technique, ADASYN [41], controls the number of generated samples by the number of sam-

ples in different classes within small groups. Again, this technique still suffers distortion of the decision boundary if the boundary region is class imbalanced. Additionally, such mentioned techniques have not utilized statistical data information. A recent work, Gaussian Distribution Based Oversampling (GDO) [96], balances data class based on the statistical information of data instead. However, its strong assumption of data distribution (data follow Gaussian) reduces the technique’s effectiveness in real data.

To alleviate the negative effects of data imbalance and avoid the drawbacks of existing techniques, we propose a minority oversampling technique that focuses on balancing the high-entropy region that provides the most critical information to the deep learning models. Besides, the technique enhances synthetic data’s chance to fall into the minority class to reduce model errors. By carefully generating synthetic data near minority samples, our proposed technique also preserves the best data topology. Besides, our technique does not need any statistical assumption.

To find informative samples, we leverage an entropy-based deep active learning technique that is able to select samples yielding high entropy to deep learning models. We denote the location of informative samples as the informative region. We then balance this region first, and the remaining data are balanced later so that it would reduce the decision distortion mentioned earlier. For each minority sample in this region, we safely generate its synthetic neighbors so that the global data topology is still preserved. However, generating synthetic samples in this region is risky because it can easily fall across the decision boundary. Therefore, we find a direction to generate synthetic samples by maximizing their posterior probability based on Bayes’s Theorem. However, maximizing the posterior probability is facing infeasible computation in the denominator. To overcome this, we maximize the posterior ratio instead so that the denominator computation can be avoided. This also ensures that the synthetic samples are not only close to the minority class but also far from the majority class. The remaining data are eventually balanced by a similar procedure.

The proposed technique alleviates the class imbalance problem. Our experiments indicate that we can achieve better classification results over widely-used techniques in all experimental cases by applying the proposed strategy.

Our work has the following main contributions:

1. Exploring the impact of class imbalance mitigations on deep learning via visualization and experiments.
2. Proposing a new minority oversampling-based technique, namely Synthetic Information towards Maximum Posterior Ratio, to balance data classes and alleviate data imbalance impacts. Our technique is enhanced by the following key points.
  - (a) Leveraging an entropy-based active learning technique to prioritize the region that needs to be balanced. It is the informative region where samples provide high information entropy to the model.
  - (b) Leveraging Maximum Posterior Ratio and Bayes’s theorem to determine the direction to generate synthetic minority samples to ensure the synthetic data fall into the minority class and not fall across the decision boundary. To our best knowledge, this is the first work utilizing the posterior ratio for tackling class imbalanced data.
  - (c) Approximating the likelihood in the posterior ratio using kernel density estimation, which can approximate a complicated topology. Thus, the proposed technique is able to work with large, distributively complex data.
  - (d) Carefully generating synthetic samples surrounding minority samples so that the global data topology is still preserved.
3. We applied our technique to 41 real datasets with a diversity of imbalance ratio and the number of features.



4. We compare our technique with different widely-used and recent techniques. The results show that the proposed technique outperforms others.

The rest of this paper is organized as follows. Section 3.2 introduces related concepts that will be used in this work, i.e., Imbalance Ratio, Macro F1-score, AUC, and Entropy-based active learning. Section 3.3 will provide more detail on the problem of learning from an imbalanced dataset. Our proposed solution to balance dataset, Synthetic Information towards Maximum Posterior Ratio, will be explained comprehensively in Section 3.5. Section 3.6 discusses the technique implementation and complexity. We will show experiments on different datasets, including artificial and real datasets in Section 3.7. We also discuss experimental results in the same section. In Section 3.4, we briefly review other existing works. Section 3.8 concludes the study and discusses future work.

## 3.2 Preliminaries

In this section, we introduce related concepts that will be used in our work.

### 3.2.1 Imbalance Ratio (IR)

For binary classification problems, we use imbalance ratio (IR) to depict the data imbalance as it has been widely used. IR is the ratio of the majority class samples to the minority class’s samples. For example, if a dataset contains 1000 class-A and 100 class-B samples, the Imbalance Ratio is 10:1.

### 3.2.2 Evaluation Metrics

In this work, we evaluate balancing data techniques by classification performance. Specifically, we use F1-Score and Area Under the Curve (AUC) as evaluation metrics. We measure the Macro-averaging for measuring F1-scores in which we compute scores per class and take the average of all classes with the same weight regardless of how often they appear in the datasets. These are fair measurements for imbalanced test datasets.

F1 score is computed based on two factors Recall and Precision as follows:

$$Recall = \frac{TP}{TP + FN} \quad (3.1)$$

$$Precision = \frac{TP}{TP + FP} \quad (3.2)$$

$$F1 - score = \frac{2 * Recall * Precision}{Recall + Precision}, \quad (3.3)$$

where  $T$  and  $F$  stand for True and False;  $P$  and  $N$  stand for Positive and Negative.

We also measure AUC [92] scores as it is an important metric to evaluate imbalanced data. AUC is derived from Receiver Operating Characteristic curve (ROC). In this work, we utilize a skit-learn library to compute AUC; the library can be found in `sklearn.metrics.auc`.

### 3.2.3 Entropy-based Active Learning

To find informative samples, we leverage entropy-based active learning. The technique gradually selects batch-by-batch samples that provide high information to the model based on information entropy theory [84]. The information entropy is quantified based on the “surprise” to the model in terms of class prediction probability. Take a binary classification, for example, if a sample is predicted to be 50% belonging to class A and 50% belonging to class B, this sample has high entropy and is informative to the model. In contrast, if it is predicted to be 100% belonging to class A, it is certain and gives zero information to the model. The class entropy  $E$  for each sample can be computed as follows.

$$E(x, \theta) = - \sum_i^n P_\theta(y = c_i|x) \log_n P_\theta(y = c_i|x) \quad (3.4)$$

where  $P_\theta(y = c_i|x)$  is the probability of data  $x$  belonging to the  $i$ th class of  $n$  classes with current model parameter  $\theta$ .

In this work, we consider a dataset containing  $N$  pairs of samples  $X$  and corresponding labels  $y$ , and a deep neural network with parameter  $\theta$ . At the first step  $t^{(0)}$ , we train the classifier with parameter  $\theta^{(0)}$  on a random batch of  $k$  labeled samples and use the  $\theta^{(0)}$  to predict the labels for the rest of the data (we assume their labels are unknown). We then compute the prediction entropy of each sample based on Equation 3.4. We are now able to collect the first batch of informative samples by selecting  $k$  samples based on the top  $k$  highest entropy. We query labels for this batch and concatenate them to existing labeled data to train the classifier parameter  $\theta^{(1)}$  in the next step  $t^{(1)}$ . Steps are repeated until the number of informative samples reaches a pre-set informative portion (IP). For example,  $IP = 0.3$  will select the top 30% high entropy samples as informative samples.

### 3.3 The Problem of Learning From Imbalanced Datasets

In this section, we review the problem of learning from imbalanced datasets. Although the problem may apply to different machine learning methods, we focus on deep learning in this work.

Figure 3.1 illustrates our problem on a binary classification. The imbalance in the informative region (light blue eclipse) could lead to classification errors. The dashed green line depicts the expected boundary, while the solid blue line is the model’s boundary. Since the minority class lacks data in this region, the majority class will dominate the model even with a few noisy poorly-placed samples, which leads to a shift of the model’s boundary. In contrast to the study by Ertekin *et al.* [27] which assumes the informative region is more balanced by nature and proposes a solution that only classifies over the informative samples, our assumption is different. We consider the case that the informative region contains highly imbalanced data, which we believe happens in most real scenarios. The problem could be more severe in a more complex setting such as high-dimensional and topologically complex data. Therefore, we proposed a technique to tackle the problem of data imbalance by over-

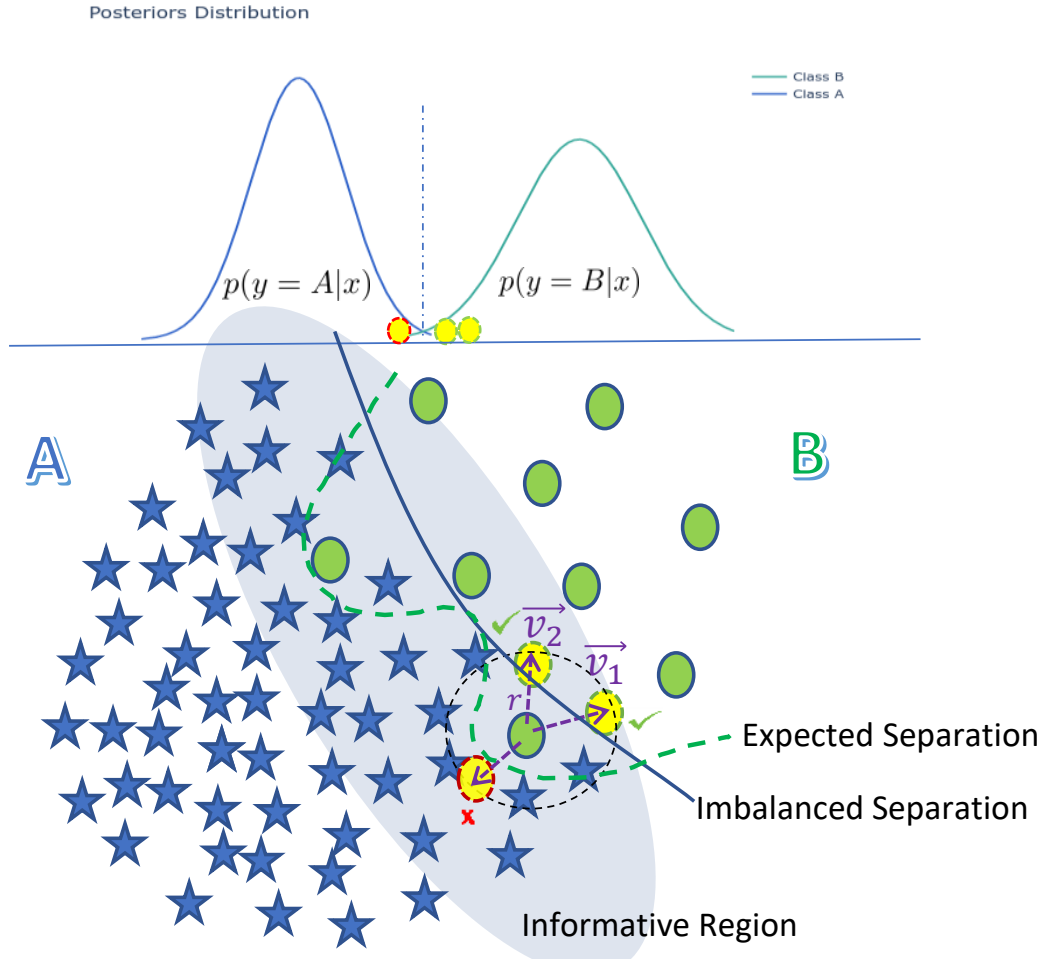


Figure 3.1: Learning from imbalanced datasets

sampling the minority class in an informative manner. The detail of our proposed technique will be described in Section 3.5.

### 3.4 Related Work

In the last few decades, many solutions have been proposed to alleviate the negative impacts of data imbalance in machine learning. However, most of them are not efficiently extended for deep learning. This section reviews algorithms to tackle class-imbalanced data

that can be extended for deep learning. These techniques can be categorized into three main categories, i.e., sampling, cost-sensitive, and ensemble learning approaches.

### 3.4.1 Sampling-based Approach.

Compared to other approaches, resampling techniques have attracted more research attention as they are independent of machine learning algorithms. This approach can be divided into two main categories, over-sampling, and under-sampling techniques. Such sampling-based methods e.g., [85], [32], [38], [56], [28] mainly generate a balanced dataset by either over-sampling the minority class or down-sampling the majority class. Some techniques are not designed for deep learning; however, we still consider them in this work since they are independent of the machine learning model architecture. In a widely used method, SMOTE [14], Chawla *et al.* attempt to oversample minority class samples by connecting a sample to its neighbors in feature space and arbitrarily drawing synthetic samples along with the connections. However, one of SMOTE drawbacks is that if there are samples in the minority class located in the majority class, it creates synthetic sample bridges toward the majority class [37]. This renders difficulties in differentiation between the two classes. Another SMOTE-based work, namely Borderline-SMOTE [39] was proposed in which its method aims to do SMOTE with only samples near the border between classes. The samples near the border are determined by the labels of its  $k$  distance-based neighbors. This "border" idea is similar to ours to some degree. However, finding a good  $k$  is critical for a heuristic machine learning algorithm such as deep learning, and it is usually highly data-dependent.

Among specific techniques for deep learning, generating synthetic samples in the minority class by sampling from data distribution is becoming more attractive as they outperform other methods in high dimensional data [61]. Regarding images, several deep learning generative-based methods have been proposed as deep learning is capable of capturing good image representations. [77] [22] [72] utilized Variational Autoencoder as a generative model

to arbitrarily generate images from learned distributions. However, most assumed simple prior distributions, such as Gaussian for minor classes, tend to simplify data distribution and might fail in more sophisticated distributions. In addition, most of the works in this approach tackle image datasets, while our proposed method focuses on tabular datasets as this is a missing piece in the field [50].

Under the down-sampling category, existing techniques mainly down-sample the majority class to balance it with the minority class. There are several proposed techniques to simplify the majority. A straightforward way is to randomly remove the majority class samples. Other works, e.g., [27], [7] find near-border samples and authors believe the imbalance ratio in these areas is much smaller than that in the entire dataset. They then classify this small pool of samples to improve the performance and expedite the training process for the SVM-based method. However, this method was only designed for SVM-based methods, which mainly depend on the support vectors. Also, this potentially discards essential information of the entire dataset because only a small pool of data is used.

### 3.4.2 Cost-sensitive Learning Approach

Cost-sensitive learning techniques usually require modifications of algorithms on the cost functions to balance each class's weight. Specifically, such cost-sensitive techniques put higher penalties on majority classes and less on minority classes to balance their contribution to the final cost. For example, [21] provided their designed formula  $(1-\beta^n)/(1-\beta)$  to compute the weight of each class based on the effective number of samples  $n$  and a hyperparameter  $\beta$  which is then applied to re-balance the loss of a convolutional neural network model. [45],[76], [67] assign classes' weights inversely proportional to sample frequency appearing in each class. Hamed et al. [68] proposed an SVM-based cost-sensitive approach (SVMCS) that uses svm with a class-weighted loss function.

### 3.4.3 Ensemble Learning Approach

Ensemble learning has achieved high performance in classification for its generalizability. Thus, it could reduce the bias due to class imbalance. Ensemble learning can be constructed by combining several base classifiers with different sampling-based approaches. In [15, 81], Chawla *et al.* and Seiffert *et al.* proposed variants of ensemble learning in which the data are balanced based on oversampling method SMOTE and then applying ensemble learning on balanced data. Similarly, authors in [60] generate cluster-based synthetic data and combine it with an evolutionary algorithm. Liu *et al.* in [63] balances the data by applying a fuzzy-based oversampling technique and building ensemble learning classifiers on this data. Zhou and Liu in [64] explore a method, namely Easy Ensemble classifier (EE), to perform ensemble learning on the random under-sampling balanced data.

## 3.5 Federated Learning for Skewed Distribution Using Sample Weights

To alleviate the negative effects of data imbalance, we propose a comprehensive approach, Synthetic Information towards Maximum Posterior Ratio (SIMPOR), which aims to generate synthetic samples for minority classes. We first find the informative region where informative samples are located and balance this region by creating surrounding synthetic neighbors for minority samples. The remaining region is then fully balanced by arbitrarily generating minority samples' neighbors. We elaborate on how our strategy is developed in the rest of this section.

### 3.5.1 Methodology Motivation

As Chazal and Michel mentioned in their work [55], the natural way to highlight the global topological structure of the data is to connect data points' neighbors; our proposed method aligns with their observation by generating surrounding synthetic neighbors for minority samples to preserve data topology. Thus, our technique not only generates more data for minority class but also preserve the underlying topological structure of the entire data.

Similar to [27] and [7], we believe that informative samples play the most important role in the prediction success of both traditional machine learning models (e.g., SVM, Naive Bayes) and modern deep learning approaches (e.g., neural network). Thus, our technique finds these informative samples and focuses on augmenting minority data in this region. In this work, we apply an entropy-based active learning strategy mentioned in 3.2.3 to find the samples that maximize entropy to the model. This strategy is perhaps the most popular active learning technique and over-performs many other techniques on several datasets [35], [75] [82].

### 3.5.2 Generating Minority Synthetic Data

A synthetic neighbor  $x'$  and its label  $y'$  can be created surrounding a minority sample  $x$  by adding a small random vector  $v$  to the sample,  $x' = x + v$ . This lays on the d-sphere surface centered by  $x$ , and the d-sphere's radius is set by the length of vector  $\vec{v}$ ,  $|\vec{v}|$ . It is, however, critical to generate synthetic data in the informative region because synthetic samples can unexpectedly jump across the decision boundary. This can be harmful to models as this might create outliers and reduce the model's performance. Therefore, we safely find vector  $\vec{v}$  towards the minority class, such as  $\vec{v}_0$  and  $\vec{v}_1$  depicted in Figure 3.1. Our technique is described via a binary classification scenario as follows.

Let's consider a binary classification problem between majority class A and minority class B. From the Bayes' theorem, the posterior probabilities  $p(y' = A|x')$  or  $p(y' = B|x')$  can be used to present the probabilities that a synthetic sample  $x'$  belongs to class A or class B, respectively. Let the two posterior probabilities be  $f_0$  and  $f_1$ ; they can be expressed as follows.

$$p(y' = A|x') = \frac{p(x'|y' = A) p(A)}{p(x')} = f_0 \quad (3.5)$$

$$p(y' = B|x') = \frac{p(x'|y' = B) p(B)}{p(x')} = f_1 \quad (3.6)$$



As mentioned earlier, we would like to generate each synthetic data  $x'$  that maximizes the probability of  $x'$  belonging to the minority class  $B$  and minimizes the chance  $x'$  falling into the majority class  $A$ . Thus, we propose a technique that maximizes the fractional posterior  $f$ ,

$$f = f_1/f_0 \tag{3.7}$$

$$= \frac{p(x'|y' = B) p(B)}{p(x'|y' = A) p(A)}. \tag{3.8}$$

We use non-parametric kernel density estimates (KDE) to approximate the likelihoods  $p(x'|y' = A)$  and  $p(x'|y' = B)$  as KDE is flexible and does not require specific assumptions about the data distribution. One can use a parametric statistical model such as Gaussian to approximate the likelihood; however, it oversimplifies the data and does not work effectively with topological complex data, especially in high dimensions. In addition, parametric models require an assumption about the distribution of data which is difficult in real-world problems since we usually do not have such information. On the other hand, KDE only needs a kernel working as a window sliding through the data. Among different commonly used kernels for KDE, we choose Gaussian Kernel as it is a powerful continuous kernel that would also ease the derivative computations for finding optima.

Additionally, we assume prior probabilities of observing samples in class A ( $p(A)$ ) and class B ( $p(B)$ ) (in Equation 3.8) are constant. Hence, these probabilities do not affect our algorithm in terms of generating synthetic neighbors for minority samples because we only determine the relative direction between the minority and the majority class. Thus, they can be canceled out at the end of the equation reduction.

We reduce Equation 3.8 as follows. Let  $X_A$  and  $X_B$  be the subsets of dataset  $X$  which contain samples of class A and class B,  $X_A = \{x : y = A\}$  and  $X_B = \{x : y = B\}$ .  $N_A$  and  $N_B$  are the numbers of samples in  $X_A$  and  $X_B$ .  $d$  is the number of data dimensions.  $h$  presents

the width parameter of the Gaussian kernel. The posterior ratio for each synthetic sample  $x'$  then can be estimated as follows:

$$f = \frac{p(x'|y' = B) p(B)}{p(x'|y' = A) p(A)} \quad (3.9)$$

$$\propto \frac{\frac{1}{N_B h^d} \sum_{i=1}^{N_B} (2\pi)^{-\frac{d}{2}} e^{\frac{1}{2}(\frac{x' - x_{B_i}}{h})^2} p(B)}{\frac{1}{N_A h^d} \sum_{j=1}^{N_A} (2\pi)^{-\frac{d}{2}} e^{\frac{1}{2}(\frac{x' - x_{A_j}}{h})^2} p(A)} \quad (3.10)$$

$$\propto \frac{N_A \sum_{i=1}^{N_B} e^{\frac{1}{2}(\frac{x' - x_{B_i}}{h})^2} p(B)}{N_B \sum_{j=1}^{N_A} e^{\frac{1}{2}(\frac{x' - x_{A_j}}{h})^2} p(A)} \quad (3.11)$$

$$\propto \frac{\sum_{i=1}^{N_B} e^{\frac{1}{2}(\frac{x' - x_{B_i}}{h})^2}}{\sum_{j=1}^{N_A} e^{\frac{1}{2}(\frac{x' - x_{A_j}}{h})^2}}. \quad (3.12)$$

Because we want to generate neighbors for each minority sample that maximizes Function  $f$  in Equation 3.12, we examine points lying on the sphere centered at the minority sample with a small radius  $r$ . As a result, we can find a vector  $\vec{v}$  so that it can be added to the sample to generate a new sample. The relationship between a synthetic sample  $x'$  and a minority sample can be described as follows,

$$\vec{x}' = \vec{x} + \vec{v}, \quad (3.13)$$

where  $|\vec{v}| = r$ , and  $r$  is sampled from a Gaussian distribution,

$$r \sim \mathcal{N}(0, (\alpha R)^2), \quad (3.14)$$

where  $\alpha R$  is the standard deviation of the Gaussian distribution and  $0 < \alpha \leq 1$ .

The range parameter  $R$  is relatively small and computed as the average distance of a minority sample  $x$  to its  $k$ -nearest neighbors. This will ensure that the generated sample will be surrounding the minority sample. The Gaussian distribution with the mean of zero and the standard deviation  $\alpha R$  controls the distance between the synthetic samples and the minority sample. The standard deviation is tuned from 0 to  $R$  by a coefficient  $\alpha \in (0, 1]$ . The larger the  $\alpha$  is, the farther synthetic data created from its original sample. Consider a minority sample  $x$  and its  $k$ -nearest neighbors in the Euclidean space,  $R$  can be computed as follows:

$$R = \frac{1}{k} \sum_1^k \|x - x_j\|, \quad (3.15)$$

where  $\|x - x_j\|$  is the Euclidean distance between a minority sample  $x$  and its  $j$ th neighbor.  $k$  is a parameter indicating selected number of neighbors.

Figure 3.2 depicts a demonstration of finding 3 synthetic samples from 3 minority samples. In fact, one minority can be re-sampled to generate more than one synthetic sample. For a minority sample  $x_0$ , we find a synthetic sample  $x'_0$  by maximizing the objective function  $f(x'_0)$ ,  $x'_0 \in X$  with a constraint that the Euclidean length of  $\vec{v}_0$  equals to a radius  $r_0$ ,  $\|\vec{v}_0\| = r_0$  or  $\|\vec{x}'_0 - \vec{x}_0\| = r_0$  (derived from Equation 3.13).

The problem can be described as a constrained optimization problem. For each minority sample  $x$ , we find a synthetic sample  $x' \in \mathbb{R}^d$  lying on the  $d$ -sphere centered at  $x$  with radius  $r$  and maximizing function in Equation 3.12,

$$\max_{x'} f(x') \quad \text{s.t.} \quad \|\vec{x}' - \vec{x}\| = r. \quad (3.16)$$

Interestingly, the problem in Equation 3.16 can be solved numerically. Function  $f(x)$  in Equation 3.12 is defined and continuous for  $x' \in (-\infty, +\infty)$  because all of the exponential components (Gaussian kernels) are continuous and greater than zero. In addition, the

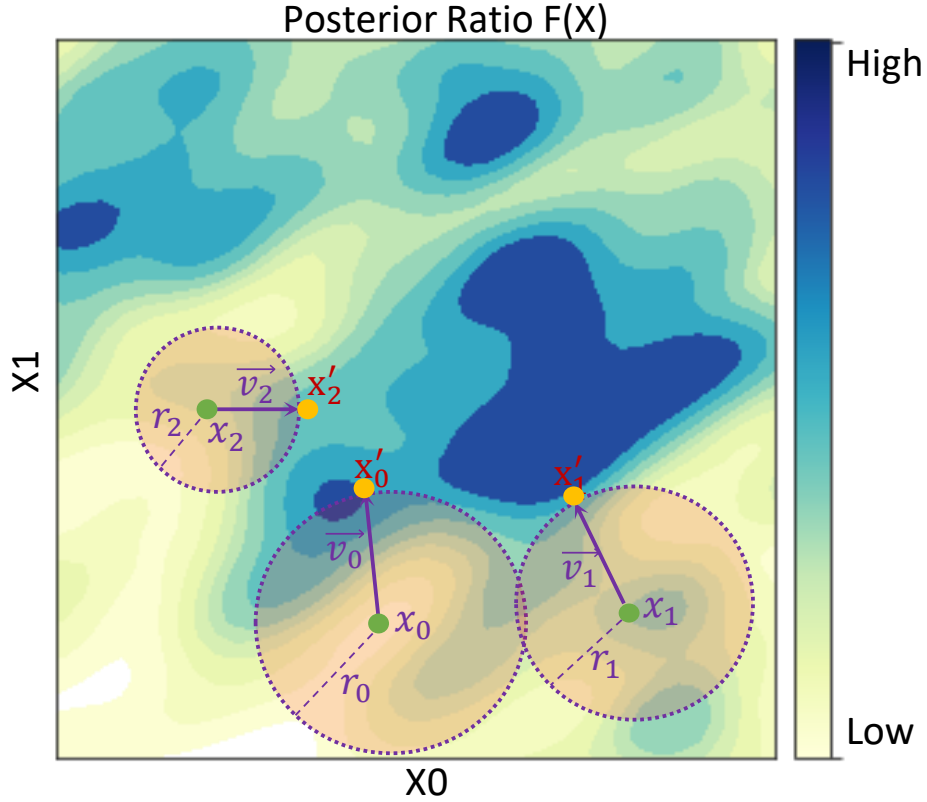


Figure 3.2: Demonstration on how SIMPOR generates three synthetic samples  $x'_0, x'_1, x'_2$ , from three minority samples  $x_0, x_1, x_2$ , by maximizing the Posterior Ratio.

constraint,  $\|\vec{x}' - \vec{x}\| = r$ , which contains all points on the sphere centered at  $x$  with radius  $r$  is a closed set ([5]). Thus, a maximum exists as proved in [4]. To enhance the diversity of synthetic data, we accept either the global maximum or any local maximum so that the synthetic samples will not simply go to the same direction.

We solve the problem in Equation 3.16 by using the Projected Gradient Ascent approach in which we iteratively update the parameter to go up the gradient of the objective function. A local maximum is found if the objective value cannot be increased by any local update. For simplification, we rewrite the problem in Equation 3.16 by shifting the origin to the

considered minority sample. The problem becomes finding the maximum of function  $f(x')$ ,  $x' \in \mathbb{R}^d$ , constrained on a d-sphere, i.e.,  $\|x'\| = r$ . Our solution can be described in Algorithm 3.1. After shifting the coordinates system, we start by sampling a random point on the constraint sphere (line 1 – 2). The gradient of the objective function at time  $t$ ,  $g_t(x'_t)$ , is computed and projected onto the sphere tangent plane as  $p_t$  (line 4–5). It is then normalized and used for update a new  $x'_{t+1}$  by rotating a small angle  $lr * \theta$  (line 6 – 7). The algorithm stops when the value of  $f(x')$  is not increased by any update of  $x'$ . We finally shift to the original coordinates and return the latest  $x'_t$ .

---

**Algorithm 3.1** Sphere-Constrained Gradient Ascent for Finding Maximum

---

**Input:** A minority sample  $x_0$ , objective function  $f(x, X)$

**Parameter:**

$r$  : The radius of the sphere centered at  $x_0$

$\theta$  : Sample space  $\theta \in [0, 2\pi]$

$lr$  : Gradient ascent learning rate

**Output:** An local maximum  $x'$

- 1: Shift the Origin to  $x_0$
  - 2: Randomly initiate  $x'_t$  on the sphere with radius  $r$
  - 3: **while** converge condition **do**
  - 4:   Compute the gradient at  $x'_t$   

$$g_t(x'_t) = \nabla f(x'_t)$$
  - 5:   Project the gradient onto the sphere tangent plane  

$$p_t = g_t - (g_t \cdot x'_t)x_t$$
  - 6:   Normalize projected vector  

$$p_t = p_t / \|p_t\|$$
  - 7:   Update  $x'$  on the constrained sphere  

$$x'_{t+1} = x'_t \cos(lr * \theta) + p_t \sin(lr * \theta)$$
  - 8: **end while**
  - 9: Shift back to the Origin
  - 10: **return**  $x'_t$
-

To reduce the chance of misplacing synthetic samples on another class region because of noisy borderline and mislabeled minority samples, we set a policy for rejecting minority candidates which are selected for oversampling. The idea is to reject candidates surrounded mainly by other class samples. More specifically, we count the labels of the candidate’s  $k$ -nearest neighbors and reject this candidate if there exists a class that its’ number of samples is greater than the number of the minority samples). For example, the candidate is rejected when a class-A sample is selected for generating synthetic data, and its 5-nearest neighbors contain four class-B samples and one class-A sample. This is to avoid selecting mislabeled samples and noisy borderline samples for oversampling.

### 3.5.3 Algorithm

Our strategy can be described in Algorithm 3.2. The algorithm takes an imbalanced dataset as its input and results in a balanced dataset which is a combination of the original dataset and synthetic samples. We first choose an active learning method  $AL(\cdot)$  and find a subset of informative samples  $\mathcal{S}$  by leveraging entropy-based active learning (lines 1–2). We then generate synthetic data to balance  $\mathcal{S}$ . For each random sample  $x_i^c$  in  $\mathcal{S}$  and belonging to minority class  $c$ , we randomly sample a small radius  $r$  and find a synthetic sample that lies on the sphere centered at  $x_i^c$  and maximizes the posterior ratio in Equation 3.12 (lines 3–11). The process is repeated until the informative set  $\mathcal{S}$  is balanced. Similarly, the remaining region is balanced, which can be described in the pseudo-code from line 12 to line 20. The final output of the algorithm is a balanced dataset  $D'$ .

## 3.6 Algorithm Time Complexity

The costly part of SIMPOR is that each synthetic sample requires computing a kernel density estimation of the entire dataset. Elaborately, let  $n$  be the number of samples of the dataset. In the worst case, the numbers of samples of minority and majority class are  $N_B = 1$  and  $N_A = n - 1$ , respectively. We need to generate  $n - 2$  synthetic samples to balance the

---

**Algorithm 3.2** SIMPOR

---

**Input:** Original Imbalance Dataset  $D$  including data  $X$  and labels  $y$ .

**Parameter:**  $MA$  is the majority class,  $MI$  is a set of other classes.

$k$ : Number of neighbors of the considered sample which determines the maximum range of the sample to its synthetic samples.

$\alpha$ : preset radius coefficient  $Count(c, P)$ : A function to count class  $c$  sample number in population  $P$ .

$G(x_0, f, r)$ : Algorithm 3.1, which returns a synthetic sample on sphere centered at  $x_0$  with radius  $r$  and maximize Equation 3.12.

**Output:** Balanced Dataset  $D'$  including  $\{X', y'\}$

- 1: Select an Active Learning Algorithm  $AL()$
  - 2: Query a subset of informative samples  $S \in D$  using  $AL$ :  $s \leftarrow AL(D)$   
{Balance the informative region}
  - 3: **for**  $c \in MI$  **do**
  - 4:   **while**  $Count(c, S) \leq Count(MA, S)$  **do**
  - 5:     Select a random  $x_i^c \in S$
  - 6:     Reject and reselect  $x_i^c$  if its label is dominated among  $k$ -nearest labels
  - 7:     Compute maximum range  $R$  based on  $k$ -nearest neighbors
  - 8:     Randomly sample a radius  $r \sim \mathcal{N}(0, \alpha R)$
  - 9:     Generate a synthetic neighbor  $x'$  from  $x_i^c$ :  $x' = G(x_i^c, f, r)$
  - 10:     Append  $x'$  to  $D'$
  - 11:   **end while**
  - 12: **end for**  
{Balance the remaining region}
  - 13: **for**  $c$  in  $MI$  **do**
  - 14:   **while**  $Count(c, D') \leq Count(MA, D')$  **do**
  - 15:     Select a random  $x_j^c \in \{X - S\}$
  - 16:     Compute maximum range  $R$  based on  $k$
  - 17:     Randomly sample a radius  $r \sim \mathcal{N}(0, \alpha R)$
  - 18:     Generate a synthetic neighbor  $x'$  of  $x_j^c$
  - 19:     Append  $x'$  to  $D'$
  - 20:   **end while**
  - 21: **end for**
  - 22: **return**
-

dataset completely. Since each generated sample must loop through the entire dataset of size  $n$  to estimate the density, the algorithm complexity is  $O(n^2)$ .

Although generating synthetic data is only a one-time process, and this does not affect the classification efficiency in the testing phase, we still try to alleviate its weakness by providing parallelized implementations to reduce the time complexity to  $O(n)$ . Specifically, each exponential component in Equation 3.12 is computed parallelly, utilizing GPU or CPU threads. Ellaborately, Equation 3.12 can be rewritten as  $N_B$  components of  $e^{\frac{1}{2}(\frac{x-X_{B_i}}{h})^2}$  and  $N_A$  components of  $e^{\frac{1}{2}(\frac{x-X_{A_i}}{h})^2}$ . Fortunately, they are all independent and can be processed parallelly. Thus, with a sufficient hardware resource, the consumption time for the kernel density estimation of each synthetic data point is then reduced by  $N_A + N_B = n$  times, which significantly simplifies the complexity to  $O(n)$ .

### 3.7 Experiments

In this section, we explore the techniques via binary classification problems on an artificial dataset (i.e., Moon) and 41 real-world datasets (i.e., KEEL, UCI, Credit Card Fraud). Samples in Moon have two features, while other datasets contain various numbers of features and imbalance ratios. Dataset details are described in Table 3.1. The implementation steps to balance datasets follow Algorithm 3.2. To evaluate our proposed balancing technique, we compare the classification performance to different widely-used and state-of-the-art techniques. More specifically, We compare SIMPOR to SMOTE [14], Borderline-SMOTE [39], ADASYN [41], Random Oversampling (ROS), Gaussian Distribution Based Oversampling (GDO) [96], SVMCS [68], EE [64]. To evaluate the classifications performance for skewed datasets, we measure widely-used metrics, i.e., F1-score and Area Under The Curve (AUC).

#### 3.7.1 Experimental Setup

This section describes the general settings and implementation details for the experimental techniques.



Table 3.1: Dataset description.

dataset	#samples	#features	IR
glass1	214	9	1.8 (138:76)
wisconsin	683	9	1.9 (444:239)
pima	768	8	1.9 (500:268)
glass0	214	9	2.1 (144:70)
yeast1	1484	8	2.5 (1055:429)
haberman	306	3	2.8 (225:81)
vehicle1	846	18	2.9 (629:217)
vehicle2	846	18	2.9 (628:218)
vehicle3	846	18	3.0 (634:212)
creditcard	1968	30	3.0 (1476:492)
glass-0-1-2-3_vs_4-5-6	214	9	3.2 (163:51)
vehicle0	846	18	3.3 (647:199)
ecoli1	336	7	3.4 (259:77)
new-thyroid1	215	5	5.1 (180:35)
new-thyroid2	215	5	5.1 (180:35)
ecoli2	336	7	5.5 (284:52)
glass6	214	9	6.4 (185:29)
yeast3	1484	8	8.1 (1321:63)
ecoli3	336	7	8.6 (301:35)
page-blocks0	5472	10	8.8 (4913:559)
yeast-2_vs_4	514	8	9.0 (463:51)
yeast-0-5-6-7-9_vs_4	528	8	9.4 (477:51)
vowel0	988	13	10.0 (898:90)
glass-0-1-6_vs_2	192	9	10.3 (175:17)
glass2	214	9	11.6 (197:17)
yeast-1_vs_7	459	7	14.3 (429:30)
glass4	214	9	15.5 (201:13)
ecoli4	336	7	15.8 (316:20)
page-blocks-1-3_vs_4	472	10	15.9 (444:28)
abalone9-18	731	8	16.4 (689:42)
yeast-1-4-5-8_vs_7	693	8	22.1 (663:30)
glass5	214	9	22.8 (205:9)
yeast-2_vs_8	482	8	23.1 (462:20)
car_eval_4	1728	21	25.6 (1663:65)
wine_quality	4898	11	25.8 (4715:183)
yeast_me2	1484	8	28.0 (1433:51)
yeast4	1484	8	28.1 (1433:51)
yeast-1-2-8-9_vs_7	947	8	30.6 (917:30)
yeast5	1484	8	32.7 (1440:44)
yeast6	1484	8	41.4 (1449:35)
abalone19	4174	8	129.4 (689:42)

### 3.7.1.1 *SIMPOR Settings*

In order to find the informative subset, we leverage entropy-based active learning. We first utilize a neural network model playing a role as a classifier to find high-entropy samples. The detailed steps are introduced in Subsection 3.2.3. The model contains two fully connected hidden layers with *relu* activation functions and 100 neurons in each layer. The output layer applies the soft-max activation function. The model is trained in a maximum of 300 epochs with an early stop option until the loss does not change after updating weights. The model is trained firstly on a random set of data; this model is then used to predict the remaining data and compute the sample entropy. We select the top 30 percent of high-entropy samples (IP=0.3) for the informative subsets. Note that the classifier for finding informative subsets differs from the classifiers for the evaluation after all balancing techniques are applied to the data.

To solve the optimization problem in Equation 3.16 for finding optima (this differs from the classification optimization for the evaluation) introduced in Section 3.5.2, we use a gradient ascent method with the gradient rate of  $1e - 5$  and the maximum iteration of 300.

### 3.7.1.2 *Evaluation Classification Settings*

Considering each imbalanced dataset as a classification problem, we use the classification testing performance for the technique comparison. Each dataset is randomly split into two parts, 80% for training and 20% for testing. The classifiers are trained on training sets after applying the techniques. The results are reported on the raw testing sets (There isn't any technique applied on the testing sets; thus, they are also possibly class imbalanced). We use F1-score and AUC for the evaluation metrics as they are suitable and widely used to evaluate imbalanced data. Reported testing results for each dataset are the averages of 5 experimental trials.

The classifiers are constructed by neural networks with the input and output sizes corresponding to the number of datasets' features and unique labels. We use the same classifier

structure (number of hidden layers, number of neurons in each layer, learning rate, optimizer) for all compared datasets. The detail of neural network implementation is described in Table 3.2. For baseline technique settings, we follow the experimental parameter sets in [96] as we share very similar datasets and comparison techniques.

Table 3.2: Classification model settings for each dataset.

Method	Parameter
SIMPOR	k_neighbors=5, r_distribtuion=Gaussian(0,1), IP=0.3
GDO	k_neighbors=5, d=1
SMOTE	k_neighbors=5, sampling_strategy='auto',random_state=None
BL-SMOTE	k_neighbors=5, sampling_strategy='auto', random_state=None
ADASYN	k_neighbors=5, sampling_strategy='auto', random_state=None
EE	#estimators=10, Estimator=AdaBoostClassifier
ROS	sampling_strategy='auto', random_state=None, shrinkage=None
Classifier	Parameter
Architecture	neuron/layer=100, #layers=3
Optimization	optimizer='adam', epochs=200, batch_size=32, learning_rate=0.1, reduce_lr_loss(factor=0.9,epsilon=1e-4,patience=5)

### 3.7.2 SIMPOR on Artificial Moon Dataset

We implement techniques on an artificial 2-dimension dataset for demonstration purposes. We first generate the balanced synthetic MOON dataset using *sklearn* package. The generated MOON contains 3000 samples labeled in two classes, and each instance has two numerical features with values ranging from 0 to 1. We then make the dataset artificially imbalanced with an Imbalance Ratio of 7:1 by randomly removing 1285 samples from one class. As a result, the training dataset becomes imbalanced, as visualized in Figure 3.3.

Figure 3.4 captures the classification for different techniques. We also visualize the model decision boundaries to provide additional information on how the classification models are affected. We use a fully connected neural network described in Table 3.2 to classify the data.

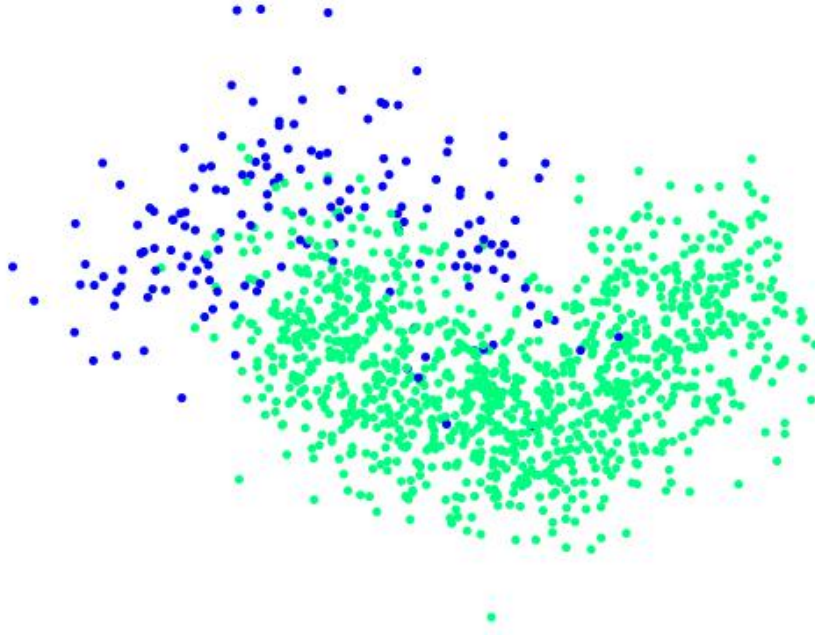


Figure 3.3: Artificial class imbalanced Moon dataset with IR of 7:1.

Table 3.3: Classification result on Moon dataset.

Metric	SIMPOR	SMOTE	BL-SMOTE	ROS	ADASYN	GDO
F1-score	0.883	0.824	0.827	0.830	0.785	0.817
AUC	0.961	0.957	0.955	0.959	0.955	0.959

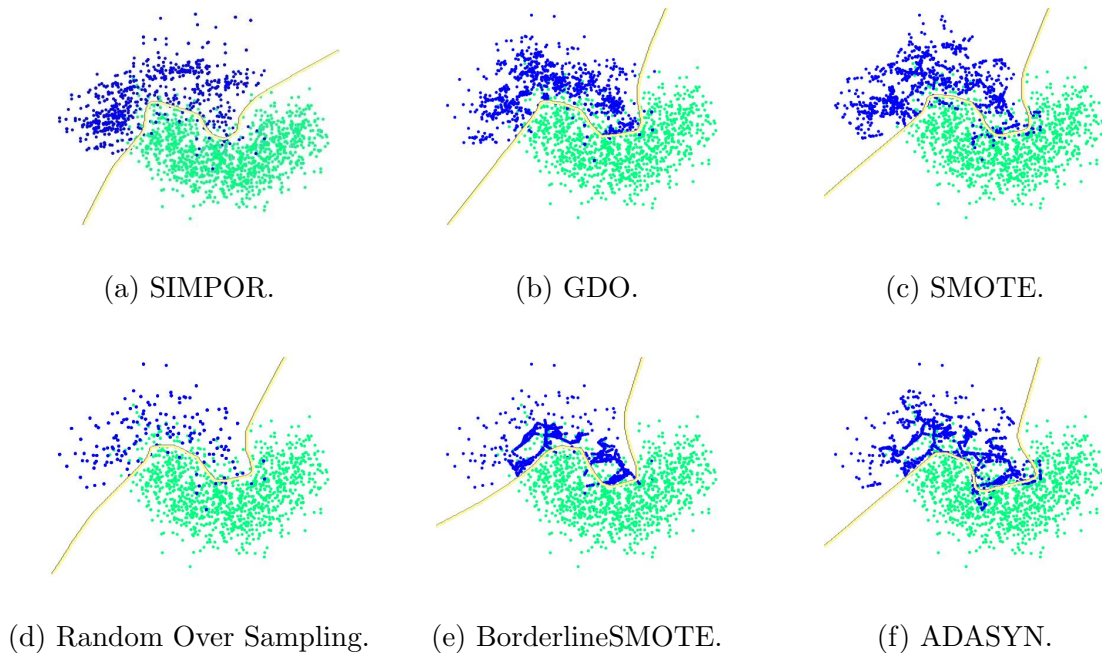


Figure 3.4: Data and model decision boundary visualization for Moon dataset.

From the visualization shown in Figure 3.4 and the classification performance results in Table 3.3, it is clear that SIMPOR performs better than others by up to 12% on F1-score and 0.6% on AUC. We can see that the Random Over Sampling technique (Figure 3.4d) which randomly duplicates minority samples might push the boundary towards the majority because the samples near the border carry significant weights. Due to the fact that SMOTE does not take the informative region into account, unbalanced data in this area lead to a severe error in decision boundary. In Figures 3.4f and 3.4e, BorderlineSMOTE (BL-SMOTE) and ADASYN focus on the area near the model’s decision boundary, but they inherit a drawback from SMOTE; any noise or mislabeled samples can, unfortunately, create very dense bridges crossing the expected border and lead to decision errors. Figure 3.4b shows that GDO also generates local gaussian groups of samples near the border and thus create errors. This phenomenon might cause by a few mis-labeled sample points. In contrast, by generating neighbors of minority samples in the direction towards the minority

class and balancing the informative region, SIMPOR (Figure 3.4a) helps the classifier to make a better decision with a solid smooth decision boundary. Poorly-placed synthetic samples are significantly less than that of others.

### 3.7.3 SIMPOR on Forty-one Real Datasets

In this section, we compare the proposed technique on 41 real two-class datasets with a variable number of features and Imbalance Ratios, i.e., KEEL datasets [3, 30], UCI datasets fetched from Sklearn tool [2, 54] and Credit Card Fraud [1] dataset. Since the original Credit Card Fraud contains a large number of banking normal and fraud transaction samples (284,807) which significantly reduces our experimental efficiency, we reduced the dataset size by randomly removing normal class transactions to reach an imbalance ratio of 3.0. Other datasets are kept as their original versions after removing bad samples (containing Null values). The datasets are described in Table 3.1.

#### 3.7.3.1 Classification Results

Table 3.4, 3.5, 3.6 and 3.7 show the classification F1-score, AUC, Precision, and Recall results, respectively. The highest scores for each dataset are highlighted in bold style. We also provide the summary of the F1 and AUC scores by “winning times” scores. We count the number of datasets for which a technique achieves the highest scores among the compared techniques and name this number “winning times”. For convention, if more than two techniques share the same highest score, the winning times will be increased for each technique. Figure 3.5 shows a summary of winning times.

As we can see from the table, the proposed technique outperforms others on both evaluation metrics, F1-score and AUC. More specifically, SIMPOR hits 24 F1-score winning times and 31 AUC winning times. Its number of F1-score winning times at 24 tripled the second winner (ADASYN) at 8, and its AUC winning times at 31 is far from the second AUC winners (GDO, EE, SVMCS) at 7.

Table 3.4: F1-score over different datasets.

	SIMPOR	GDO	SMOTE	BL-SMOTE	ADASYN	ROS	SVMCS	EE
glass1	<b>0.746</b>	0.694	0.709	0.720	0.702	0.699	0.724	0.705
wisconsin	0.967	<b>0.972</b>	0.966	0.963	0.965	0.967	0.964	0.962
pima	<b>0.754</b>	0.733	0.716	0.728	0.737	0.731	0.740	0.749
glass0	<b>0.817</b>	0.786	0.798	0.794	0.776	0.778	0.801	<b>0.817</b>
yeast1	<b>0.718</b>	0.689	0.700	0.699	0.689	0.697	0.702	0.705
haberman	0.591	0.614	0.617	0.611	<b>0.625</b>	0.585	0.600	0.608
vehicle1	0.759	0.784	0.790	<b>0.791</b>	0.789	0.773	0.754	0.768
vehicle2	<b>0.973</b>	0.933	0.938	0.968	<b>0.973</b>	0.958	0.971	0.970
vehicle3	<b>0.759</b>	0.758	0.742	0.753	0.730	0.750	0.738	0.749
creditcard	<b>0.953</b>	0.942	0.947	0.944	0.943	0.945	0.952	0.950
glass-0-1-2-3_vs_4-5-6	0.916	0.922	0.926	0.925	<b>0.927</b>	0.919	0.909	0.919
vehicle0	0.960	0.948	0.954	0.962	0.968	0.967	<b>0.972</b>	0.960
ecoli1	0.835	0.827	0.835	0.821	0.819	0.818	<b>0.849</b>	0.838
new-thyroid1	0.957	0.956	0.957	0.957	<b>0.963</b>	<b>0.963</b>	0.957	0.957
new-thyroid2	0.940	<b>0.978</b>	0.934	0.940	0.940	0.940	0.934	0.934
ecoli2	<b>0.908</b>	0.864	0.902	0.866	0.865	0.882	0.900	0.900
glass6	<b>0.926</b>	0.888	0.870	0.908	0.896	0.880	0.894	0.905
yeast3	<b>0.878</b>	0.833	0.856	0.846	0.843	0.854	0.870	0.877
ecoli3	<b>0.847</b>	0.811	0.826	0.798	0.733	0.815	0.832	0.840
page-blocks0	<b>0.907</b>	0.894	0.897	0.894	0.880	0.906	0.890	<b>0.907</b>
yeast-2_vs_4	<b>0.881</b>	0.842	0.868	0.859	0.861	0.878	0.795	0.786
yeast-0-5-6-7-9_vs_4	<b>0.802</b>	0.763	0.718	0.801	0.753	0.742	0.771	0.782
vowel0	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>
glass-0-1-6_vs_2	0.661	0.707	0.686	0.694	<b>0.740</b>	0.712	0.482	0.539
glass2	0.638	0.710	0.775	0.757	<b>0.789</b>	0.743	0.621	0.615
yeast-1_vs_7	<b>0.784</b>	0.681	0.619	0.638	0.609	0.613	0.770	0.738
glass4	<b>0.904</b>	0.845	0.855	0.853	0.855	0.853	0.853	0.865
ecoli4	<b>0.900</b>	0.857	0.872	0.880	0.865	0.872	0.888	0.888
page-blocks-1-3_vs_4	0.974	0.948	0.965	0.971	0.977	0.959	<b>0.992</b>	<b>0.992</b>
abalone9-18	0.716	0.729	0.735	0.735	0.751	0.748	0.792	<b>0.805</b>
yeast-1-4-5-8_vs_7	0.489	0.620	0.571	0.585	0.615	<b>0.630</b>	0.489	0.489
glass5	<b>0.927</b>	0.909	0.906	0.855	0.824	<b>0.927</b>	0.665	0.664
yeast-2_vs_8	<b>0.832</b>	0.703	0.712	0.708	0.691	0.747	<b>0.832</b>	<b>0.832</b>
car_eval_4	<b>1.000</b>	0.975	0.995	0.990	0.995	0.995	<b>1.000</b>	0.997
wine_quality	<b>0.743</b>	0.680	0.665	0.660	0.660	0.677	0.673	0.659
yeast_me2	0.697	0.695	0.707	0.686	0.686	0.657	<b>0.730</b>	0.717
yeast4	<b>0.743</b>	0.693	0.650	0.682	0.668	0.677	0.731	0.740
yeast-1-2-8-9_vs_7	<b>0.762</b>	0.637	0.604	0.615	0.618	0.608	0.738	0.731
yeast5	0.843	0.844	0.889	0.888	<b>0.892</b>	0.882	0.827	0.830
yeast6	0.767	0.748	0.684	<b>0.774</b>	0.741	0.738	0.758	0.763
abalone19	0.498	0.517	<b>0.541</b>	0.524	0.537	0.533	0.498	0.498

Table 3.5: AUC result over different datasets.

	SIMPOR	GDO	SMOTE	BL-SMOTE	ADASYN	ROS	SVMCS	EE
glass1	<b>0.817</b>	0.794	0.801	0.776	0.779	0.794	0.795	0.783
wisconsin	<b>0.996</b>	0.995	0.995	0.993	0.994	0.995	0.995	<b>0.996</b>
pima	<b>0.849</b>	0.825	0.818	0.811	0.816	0.820	0.832	0.834
glass0	<b>0.897</b>	0.878	0.877	0.870	0.866	0.873	0.882	0.882
yeast1	<b>0.810</b>	0.777	0.785	0.784	0.770	0.785	0.799	0.801
haberman	<b>0.724</b>	0.680	0.680	0.669	0.678	0.676	0.699	0.693
vehicle1	0.901	0.903	0.907	0.901	0.895	0.893	<b>0.911</b>	0.909
vehicle2	<b>0.998</b>	0.994	0.972	0.994	0.997	<b>0.998</b>	0.997	0.996
vehicle3	<b>0.892</b>	0.869	0.880	0.876	0.841	0.877	0.879	0.876
creditcard	<b>0.974</b>	0.973	0.967	0.963	0.969	0.966	0.971	0.972
glass-0-1-2-3_vs_4-5-6	<b>0.992</b>	0.990	0.985	0.986	0.976	0.990	0.988	0.988
vehicle0	0.994	0.993	0.993	0.994	0.993	0.994	<b>0.995</b>	0.994
ecoli1	<b>0.956</b>	0.949	0.954	0.945	0.948	0.940	0.954	0.953
new-thyroid1	0.997	<b>0.999</b>	0.998	0.998	0.997	0.997	0.998	0.998
new-thyroid2	0.998	<b>0.999</b>	0.998	0.998	0.998	0.998	<b>0.999</b>	<b>0.999</b>
ecoli2	<b>0.959</b>	0.951	0.955	0.936	0.948	0.955	0.949	0.953
glass6	0.905	<b>0.956</b>	0.925	0.876	0.890	0.936	0.926	0.946
yeast3	<b>0.974</b>	0.960	0.963	0.956	0.954	0.953	<b>0.974</b>	<b>0.974</b>
ecoli3	<b>0.902</b>	0.892	0.886	0.883	0.833	0.889	0.885	0.895
page-blocks0	0.986	<b>0.988</b>	0.983	0.984	0.985	0.987	0.983	0.983
yeast-2_vs_4	<b>0.982</b>	0.975	0.958	0.969	0.948	0.961	0.949	0.930
yeast-0-5-6-7-9_vs_4	<b>0.918</b>	0.914	0.864	0.908	0.879	0.815	0.891	0.892
vowel0	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>
glass-0-1-6_vs_2	0.868	0.849	0.892	0.871	0.893	<b>0.895</b>	0.870	0.885
glass2	0.903	0.903	0.918	0.906	<b>0.924</b>	0.903	0.900	0.897
yeast-1_vs_7	<b>0.870</b>	0.816	0.738	0.783	0.742	0.755	0.825	0.817
glass4	<b>0.992</b>	0.975	0.977	0.979	0.969	0.957	0.985	0.982
ecoli4	<b>0.988</b>	0.979	0.985	0.981	0.981	0.946	0.986	0.986
page-blocks-1-3_vs_4	0.998	<b>0.999</b>	<b>0.999</b>	<b>0.999</b>	<b>0.999</b>	0.998	<b>0.999</b>	<b>0.999</b>
abalone9-18	0.898	0.925	<b>0.935</b>	0.919	0.932	0.921	0.916	0.921
yeast-1-4-5-8_vs_7	<b>0.801</b>	0.770	0.692	0.705	0.704	0.741	0.746	0.747
glass5	<b>1.000</b>	0.996	0.995	0.988	0.990	0.951	0.995	0.995
yeast-2_vs_8	<b>0.823</b>	0.737	0.783	0.789	0.772	0.763	0.797	0.787
car_eval_4	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>
wine_quality	<b>0.859</b>	0.802	0.752	0.787	0.756	0.782	0.826	0.825
yeast_me2	<b>0.920</b>	0.903	0.875	0.864	0.848	0.871	0.896	0.905
yeast4	<b>0.907</b>	0.893	0.805	0.836	0.821	0.809	0.854	0.858
yeast-1-2-8-9_vs_7	<b>0.823</b>	0.767	0.709	0.729	0.710	0.752	0.784	0.770
yeast5	<b>0.993</b>	0.991	<b>0.993</b>	0.992	<b>0.993</b>	<b>0.993</b>	0.992	<b>0.993</b>
yeast6	<b>0.968</b>	0.940	0.851	0.944	0.935	0.936	0.956	0.952
abalone19	<b>0.806</b>	0.640	0.672	0.675	0.683	0.669	0.780	0.784



Table 3.6: Precision results over 41 datasets.

	SIMPOR	GDO	SMOTE	BL-SMOTE	ADASYN	ROS	SVMCS	EE
glass1	<b>0.755</b>	0.695	0.714	0.720	0.701	0.699	0.729	0.714
wisconsin	<b>0.965</b>	<b>0.965</b>	0.962	0.959	0.961	0.964	0.963	0.960
pima	<b>0.762</b>	0.727	0.713	0.722	0.731	0.727	0.747	0.755
glass0	<b>0.816</b>	0.774	0.790	0.780	0.764	0.766	0.796	0.813
yeast1	<b>0.734</b>	0.676	0.688	0.685	0.676	0.688	0.722	0.725
haberman	0.628	0.607	0.614	0.600	0.614	0.581	0.643	<b>0.646</b>
vehicle1	0.760	0.767	0.778	<b>0.785</b>	0.774	0.756	0.774	0.784
vehicle2	0.967	0.916	0.955	0.969	0.968	0.948	<b>0.970</b>	0.964
vehicle3	<b>0.768</b>	0.733	0.736	0.739	0.719	0.738	0.745	0.751
creditcard	0.959	0.945	0.952	0.950	0.951	0.956	<b>0.971</b>	0.964
glass-0-1-2-3_vs_4-5-6	0.911	0.894	<b>0.924</b>	<b>0.924</b>	0.921	0.913	0.909	0.920
vehicle0	<b>0.966</b>	0.926	0.951	0.952	0.965	0.955	0.965	0.962
ecoli1	0.845	0.819	0.815	0.795	0.799	0.803	<b>0.848</b>	0.840
new-thyroid1	0.960	0.940	0.960	0.960	<b>0.963</b>	<b>0.963</b>	0.960	0.960
new-thyroid2	0.963	<b>0.976</b>	0.961	0.963	0.963	0.963	0.961	0.961
ecoli2	<b>0.924</b>	0.839	0.903	0.851	0.851	0.867	0.915	0.915
glass6	0.976	0.926	0.929	<b>0.984</b>	0.959	0.949	0.982	<b>0.984</b>
yeast3	<b>0.885</b>	0.778	0.819	0.814	0.821	0.815	0.871	0.883
ecoli3	0.856	0.748	0.773	0.739	0.679	0.763	0.847	<b>0.859</b>
page-blocks0	0.927	0.846	0.865	0.844	0.824	0.865	<b>0.929</b>	0.921
yeast-2_vs_4	<b>0.897</b>	0.821	0.873	0.864	0.865	0.894	0.739	0.803
yeast-0-5-6-7-9_vs_4	<b>0.839</b>	0.715	0.731	0.775	0.767	0.726	0.822	0.833
vowel0	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>
glass-0-1-6_vs_2	0.713	0.636	0.661	0.671	0.730	<b>0.750</b>	0.465	0.535
glass2	0.692	0.643	0.718	0.701	<b>0.724</b>	0.696	0.642	0.625
yeast-1_vs_7	<b>0.944</b>	0.631	0.589	0.623	0.577	0.601	0.932	0.904
glass4	<b>0.945</b>	0.784	0.875	0.898	0.875	0.898	0.942	0.918
ecoli4	<b>0.902</b>	0.795	0.848	0.862	0.838	0.848	0.877	0.877
page-blocks-1-3_vs_4	<b>0.997</b>	0.908	0.937	0.948	0.959	0.927	0.986	0.986
abalone9-18	0.837	0.658	0.650	0.672	0.685	0.676	<b>0.942</b>	0.933
yeast-1-4-5-8_vs_7	0.479	0.571	0.548	0.561	0.566	<b>0.581</b>	0.479	0.479
glass5	<b>0.995</b>	0.852	0.961	0.869	0.842	0.971	0.687	0.687
yeast-2_vs_8	<b>0.883</b>	0.674	0.656	0.688	0.629	0.736	<b>0.883</b>	<b>0.883</b>
car_eval_4	<b>1.000</b>	0.954	0.990	0.981	0.990	0.990	<b>1.000</b>	<b>1.000</b>
wine_quality	<b>0.809</b>	0.658	0.673	0.650	0.674	0.684	0.767	0.719
yeast_me2	0.788	0.609	0.662	0.661	0.657	0.620	<b>0.922</b>	0.887
yeast4	0.859	0.614	0.604	0.637	0.613	0.632	0.815	<b>0.888</b>
yeast-1-2-8-9_vs_7	<b>0.989</b>	0.580	0.564	0.602	0.577	0.592	0.988	0.988
yeast5	0.835	0.748	0.833	0.840	0.835	0.816	<b>0.859</b>	0.836
yeast6	0.827	0.645	0.616	0.740	0.658	0.691	<b>0.831</b>	0.810
abalone19	0.497	0.510	0.513	0.506	0.506	<b>0.517</b>	0.497	0.497

Table 3.7: Recall results over 41 datasets.

	SIMPOR	GDO	SMOTE	BL-SMOTE	ADASYN	ROS	SVMCS	EE
glass1	<b>0.738</b>	0.693	0.705	0.719	0.702	0.698	0.718	0.697
wisconsin	0.969	<b>0.978</b>	0.969	0.967	0.969	0.970	0.966	0.964
pima	<b>0.746</b>	0.740	0.720	0.734	0.744	0.735	0.734	0.742
glass0	0.818	0.799	0.807	0.809	0.789	0.792	0.807	<b>0.821</b>
yeast1	0.702	0.702	<b>0.713</b>	<b>0.713</b>	0.702	0.706	0.683	0.685
haberman	0.559	0.622	0.621	0.623	<b>0.638</b>	0.590	0.565	0.578
vehicle1	0.759	0.802	0.803	0.798	<b>0.806</b>	0.792	0.738	0.755
vehicle2	0.978	0.951	0.926	0.967	<b>0.979</b>	0.968	0.972	0.975
vehicle3	0.751	<b>0.785</b>	0.749	0.768	0.742	0.762	0.732	0.750
creditcard	<b>0.947</b>	0.940	0.942	0.938	0.936	0.935	0.933	0.936
glass-0-1-2-3_vs_4-5-6	0.924	<b>0.951</b>	0.929	0.927	0.935	0.925	0.909	0.919
vehicle0	0.954	0.972	0.957	0.973	0.970	<b>0.980</b>	0.979	0.959
ecoli1	0.828	0.837	<b>0.857</b>	0.849	0.841	0.837	0.851	0.838
new-thyroid1	0.953	<b>0.972</b>	0.953	0.953	0.963	0.963	0.953	0.953
new-thyroid2	0.920	<b>0.981</b>	0.910	0.920	0.920	0.920	0.910	0.910
ecoli2	0.893	0.891	<b>0.902</b>	0.883	0.882	0.897	0.885	0.885
glass6	<b>0.888</b>	0.854	0.819	0.843	0.841	0.821	0.823	0.840
yeast3	0.871	<b>0.897</b>	0.895	0.881	0.869	0.896	0.869	0.871
ecoli3	0.841	<b>0.886</b>	<b>0.886</b>	0.867	0.797	0.875	0.822	0.824
page-blocks0	0.888	0.948	0.932	0.951	0.945	<b>0.952</b>	0.856	0.894
yeast-2_vs_4	0.867	<b>0.868</b>	0.865	0.856	0.858	0.865	<b>0.868</b>	0.805
yeast-0-5-6-7-9_vs_4	0.770	0.827	0.706	<b>0.832</b>	0.763	0.759	0.728	0.740
vowel0	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>
glass-0-1-6_vs_2	0.625	<b>0.807</b>	0.725	0.727	0.760	0.709	0.500	0.545
glass2	0.602	0.797	0.853	0.833	<b>0.875</b>	0.803	0.607	0.607
yeast-1_vs_7	0.678	<b>0.744</b>	0.654	0.656	0.647	0.626	0.662	0.628
glass4	0.873	<b>0.921</b>	0.848	0.830	0.848	0.830	0.786	0.832
ecoli4	0.904	<b>0.936</b>	0.900	0.901	0.899	0.900	0.902	0.902
page-blocks-1-3_vs_4	0.952	0.993	0.996	0.996	0.997	0.995	<b>0.999</b>	<b>0.999</b>
abalone9-18	0.631	0.819	<b>0.846</b>	0.814	0.835	0.842	0.693	0.716
yeast-1-4-5-8_vs_7	0.500	0.689	0.600	0.615	0.688	<b>0.697</b>	0.500	0.500
glass5	0.875	<b>0.988</b>	0.865	0.848	0.813	0.898	0.650	0.648
yeast-2_vs_8	0.797	0.778	<b>0.802</b>	0.767	0.783	0.790	0.797	0.797
car_eval_4	<b>1.000</b>	0.998	<b>1.000</b>	0.999	<b>1.000</b>	<b>1.000</b>	<b>1.000</b>	0.994
wine_quality	0.689	<b>0.704</b>	0.657	0.672	0.647	0.672	0.602	0.608
yeast_me2	0.627	<b>0.811</b>	0.763	0.714	0.719	0.706	0.610	0.609
yeast4	0.663	<b>0.795</b>	0.705	0.735	0.734	0.732	0.667	0.643
yeast-1-2-8-9_vs_7	0.620	<b>0.710</b>	0.652	0.629	0.667	0.630	0.589	0.580
yeast5	0.862	<b>0.972</b>	0.959	0.951	0.963	0.965	0.819	0.836
yeast6	0.735	<b>0.894</b>	0.777	0.812	0.854	0.805	0.706	0.730
abalone19	0.500	0.525	0.574	0.546	<b>0.581</b>	0.552	0.500	0.500

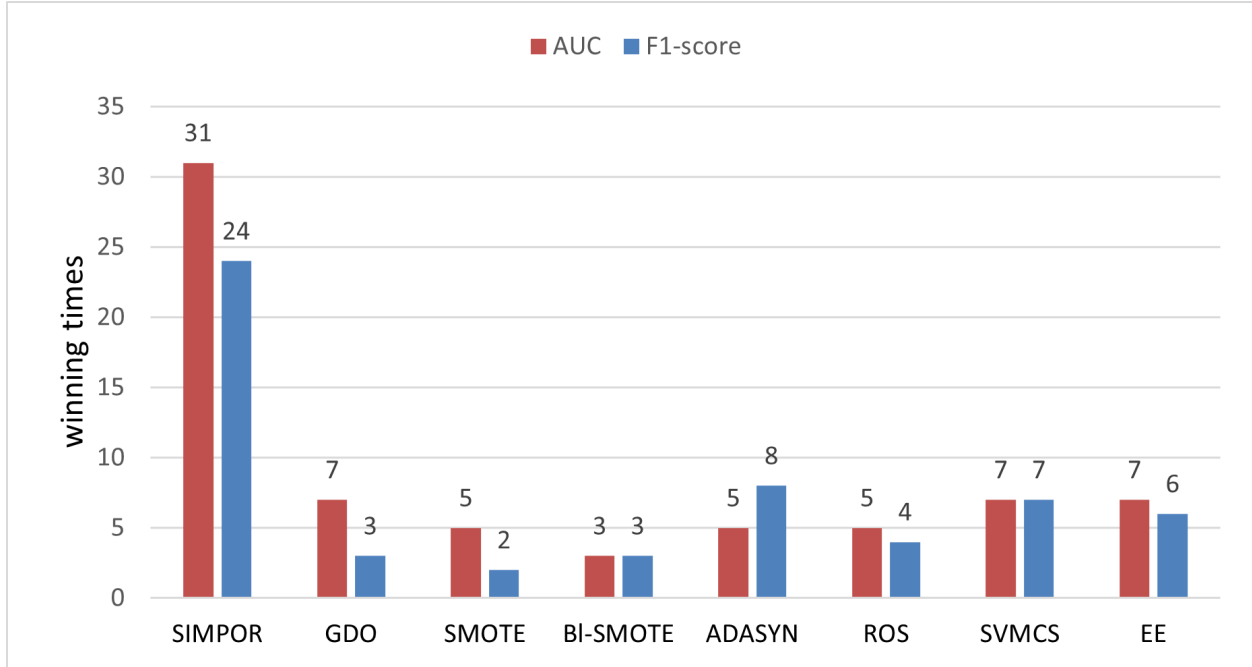


Figure 3.5: Winning times over 41 datasets.

### 3.7.3.2 Statistical Test

To further evaluate the effectiveness of the technique, we also performed a Wilcoxon Signed Rank Test [20] on the 41 dataset results (F1 score and AUC). Wilcoxon hypothesis test is relevant to our study as it is a non-parametric statistical test and does not require a specific distribution assumption for the results. On the other hand, 41 data points (corresponding to 41 datasets results) are sufficient to support this test. Our null hypothesis is that the difference between the proposed technique results and those of the other technique is insignificant. Wilcoxon signed-rank test outputs are computed over the 41 dataset results and return a p-value for each technique pair. We then compare the p-value with the significant value  $\alpha = 0.05$ . Suppose the p-value is smaller than  $\alpha$ . In that case, the evidence is sufficient to reject the hypothesis, which means the proposed technique does make a significant difference from the others, and vice versa. Table 3.8 shows the Wilcoxon p-value results.

Table 3.8: Wilcoxon signed rank hypothesis test results.

SIMPOR vs.	p-value	
	F1-score	AUC
GDO	4.96E-03	4.99E-05
SMOTE	2.32E-02	1.27E-04
BL-SMOTE	2.90E-02	3.06E-06
ADASYN	3.30E-02	1.35E-05
ROS	1.39E-02	5.58E-05
SVMCS	1.44E-03	5.80E-05
EE	2.82E-03	3.33E-04

As we can see from Table 3.8, the p-values are all smaller than the critical value of 0.05. Thus, the null hypothesis can be rejected as the supporting evidence is sufficient. In other words, the statistical result shows that the proposed technique makes a significant improvement compared to others.

### 3.7.3.3 Data Visualization

To explore more on how the techniques perform, we visualize the generated data by projecting them onto lower dimension space (i.e., one and two dimensions) using the Principle Component Analysis technique (PCA) [31]. Data’s 2-Dimension (2D) plots and 1-Dimension histograms are presented along with a hard-to-differentiate ratio (HDR) for each technique. A hard-to-differentiate ratio is defined as the ratio of intersection between 2 classes in the 1D histogram to the total of minority samples ( $HDR = \frac{\text{No. Intersection samples}}{\text{No. Minority samples}}$ ). This ratio is expected to be as small as 0% if the two classes are well separated; in contrast, 100% indicates that the two classes are unable to be distinguished in the projected 1D space. Other than HDR, we show the absolute numbers of Minority, Majority, and Intersection samples for each technique in the bottom tables. From the plots, we observe how the data are distributed in 2D space and quantify samples that are hard to be differentiated in the 1D space histograms.

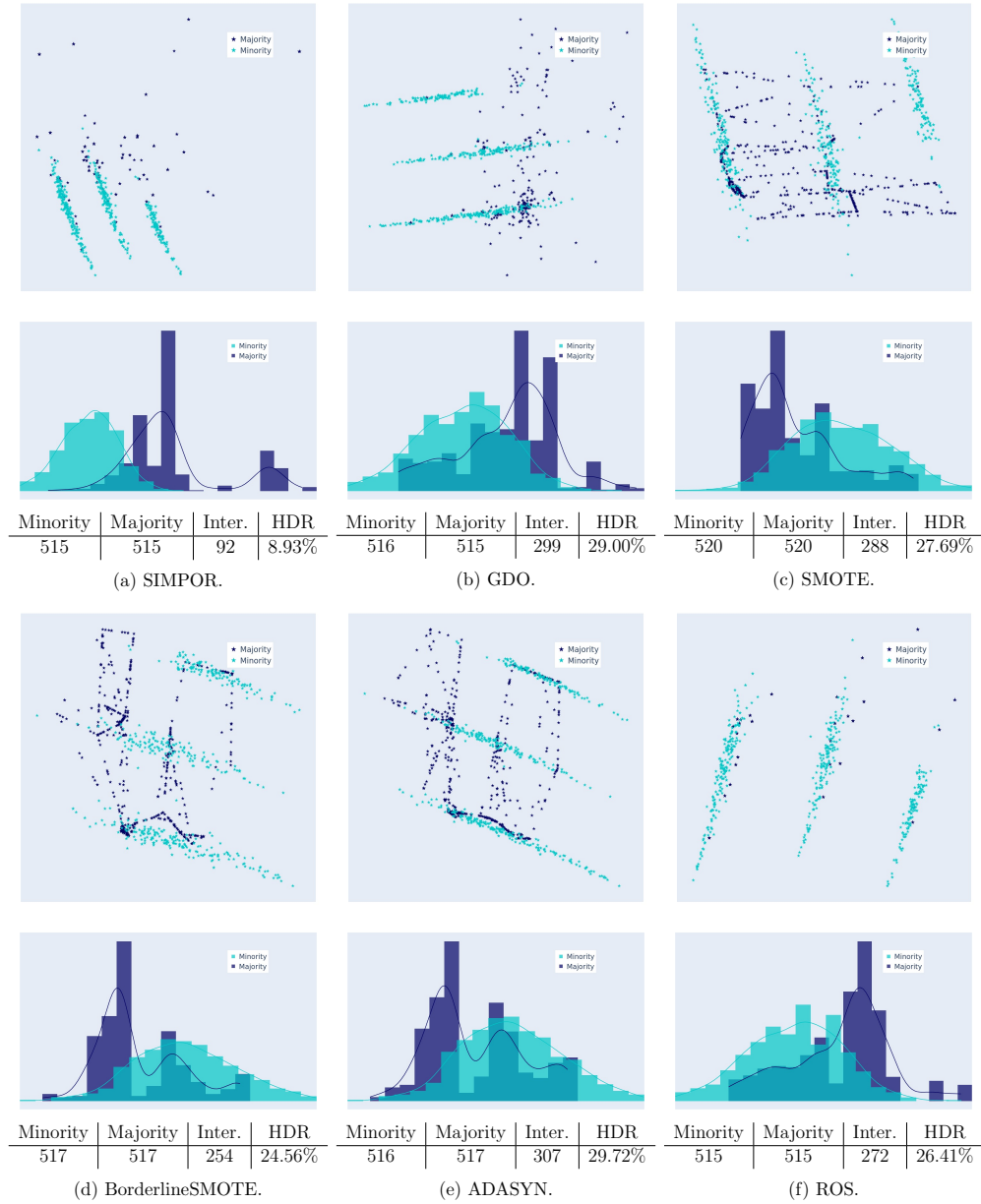


Figure 3.6: Abalone9-18: Generated training data projected onto 2-dimension space and their histograms in 1-Dimension space using Principle Component Analysis dimension reduction technique. The bottom tables illustrate the number of samples in two classes, 1-Dimension histogram intersection between 2 classes, and the hard-to-differentiate ratio between the number of intersectional samples to the number of minority samples ( $HDR = \frac{Inter.}{Minority} 100\%$ ).

To save space, we only show the plot of one dataset (i.e., Abalone9-18 dataset) in Figure 3.6. Many other datasets are observed to have similar patterns. We observe that the proposed technique does not poorly generate synthetic samples as many as other techniques do. HDR results show that SIMPOR achieves the least number of hard-to-differentiate ratio at 8.93%. As shown in the 2D visualization sub-figures, other techniques poorly place synthetic data crossed the other class. This causes by outliers or noises near the border between the two classes that other techniques do not pay attention to and mistakenly create more noise. In contrast, SIMPOR safely produces synthetic data towards the minority class by maximizing the posterior ratio; thus, it can reduce the number of poorly-placed samples.

#### 3.7.3.4 Processing Time

To explore more on how the techniques perform, we record the data processing time of resampling-based methods over 41 datasets. We don't compare to the other approaches, i.e., cost-sensitive learning and ensemble learning, because they only need negligible data processing time as they focus on classifiers other than improving the data. The processing time was recorded from our machine, which uses an Intel i7 32-thread processor and two NVIDIA 3090 Ti GPUs. Table 3.9 shows the recorded processing time over 41 datasets. Overall, our technique takes longer than other techniques as we have to compute the kernel estimation for each data point as mentioned in Section 3.6. From the table, GDO is the second slower technique, and ROS is the fastest one among compared ones. In other words, the proposed technique is slower, but it provides better F1 and AUC scores than others.

#### 3.7.4 Empirical Study on the Impact of Radius Factor $r$

In this section, we study how the classification performance is impacted by different generation radius factor  $r$  in Equation 3.14. The classification performance is measured under different distribution settings of the radius  $r$  as it controls how far synthetic data are generated from its original minority sample. We use different parameters for the Gaussian

Table 3.9: Processing time over 41 datasets.

	SIMPOR	GDO	SMOTE	BL-SMOTE	ADASYN	ROS
glass1	0.1147	0.0576	0.0020	0.0033	0.0032	0.0007
wisconsin	2.0805	0.1769	0.0024	0.0044	0.0046	0.0007
pima	0.2032	0.2066	0.0025	0.0049	0.0050	0.0006
glass0	0.2157	0.0553	0.0023	0.0035	0.0036	0.0009
yeast1	0.2457	0.4749	0.0035	0.0108	0.0104	0.0008
haberman	0.0517	0.1560	0.0022	0.0033	0.0036	0.0008
vehicle1	0.4365	0.1237	0.0025	0.0059	0.0059	0.0007
vehicle2	6.2913	0.1512	0.0029	0.0053	0.0061	0.0010
vehicle3	0.2821	0.1237	0.0024	0.0060	0.0061	0.0007
creditcard	2.1200	0.3783	0.0087	0.0184	0.0182	0.0017
glass-0-1-2-3_vs_4-5-6	0.3376	0.0459	0.0023	0.0035	0.0035	0.0008
vehicle0	7.3645	0.1198	0.0024	0.0054	0.0058	0.0007
ecoli1	0.0418	0.0337	0.0010	0.0018	0.0017	0.0004
new-thyroid1	0.5352	0.0304	0.0015	0.0024	0.0024	0.0006
new-thyroid2	0.3881	0.0359	0.0025	0.0033	0.0031	0.0009
ecoli2	0.2516	0.0266	0.0011	0.0017	0.0016	0.0004
glass6	0.3196	0.0268	0.0014	0.0025	0.0023	0.0006
yeast3	0.1374	0.2422	0.0023	0.0060	0.0059	0.0009
ecoli3	0.0658	0.0378	0.0015	0.0025	0.0024	0.0006
page-blocks0	7.9654	2.0918	0.0045	0.0143	0.0138	0.0015
yeast-2_vs_4	2.4310	0.0624	0.0017	0.0028	0.0028	0.0007
yeast-0-5-6-7-9_vs_4	0.0868	0.0632	0.0016	0.0029	0.0027	0.0007
vowel0	4.7675	0.1312	0.0018	0.0039	0.0037	0.0008
glass-0-1-6_vs_2	0.0482	0.0207	0.0013	0.0023	0.0022	0.0006
glass2	0.0501	0.0227	0.0013	0.0024	0.0024	0.0006
yeast-1_vs_7	0.4697	0.0420	0.0017	0.0026	0.0026	0.0007
glass4	0.1141	0.0197	0.0012	0.0024	0.0023	0.0006
ecoli4	0.1087	0.0310	0.0015	0.0024	0.0024	0.0006
page-blocks-1-3_vs_4	1.8742	0.0445	0.0015	0.0027	0.0026	0.0007
abalone9-18	2.9722	0.0716	0.0015	0.0028	0.0026	0.0006
yeast-1-4-5-8_vs_7	0.0881	0.0673	0.0017	0.0031	0.0028	0.0006
glass5	0.2815	0.0241	0.0017	0.0033	0.0036	0.0008
yeast-2_vs_8	0.1239	0.0441	0.0016	0.0027	0.0028	0.0007
car_eval_4	0.4381	0.1746	0.0026	0.0066	0.0049	0.0012
wine_quality	0.1622	0.8587	0.0030	0.0144	0.0137	0.0015
yeast_me2	0.1060	0.1379	0.0018	0.0042	0.0039	0.0008
yeast4	0.1083	0.1386	0.0018	0.0041	0.0039	0.0008
yeast-1-2-8-9_vs_7	0.0924	0.0757	0.0017	0.0031	0.0030	0.0008
yeast5	0.1188	0.1312	0.0019	0.0037	0.0040	0.0008
yeast6	0.0613	0.1419	0.0018	0.0037	0.0036	0.0008
abalone19	0.0890	0.3161	0.0022	0.0053	0.0054	0.0012

distribution  $\mathcal{N}(\mu, (\alpha R)^2)$ . Particularly, we fix the mean value to zero and change  $\alpha$  from 0.2 to 1 with steps of 0.2 so that the Gaussian standard deviation  $\alpha R$  will range from  $0.2R$  to  $R$ . To save space, we arbitrarily select 5 datasets to conduct this experiment. The classification results are shown in Figure 3.7.

The result figure shows that the classification performance is not very sensitive to the  $\alpha$  factor with the radius distribution standard deviation between  $0.6R$  and  $R$ . While there are only small changes within the  $\alpha$  range from 0.6 to 1, the performance is increasing in the range from 0.2 to 0.6 (i.e., *ecoli1*, *abalone9-18*, *yeast4*). These observations suggest us to use  $\alpha$  from 0.6 to 1 for the selected datasets.

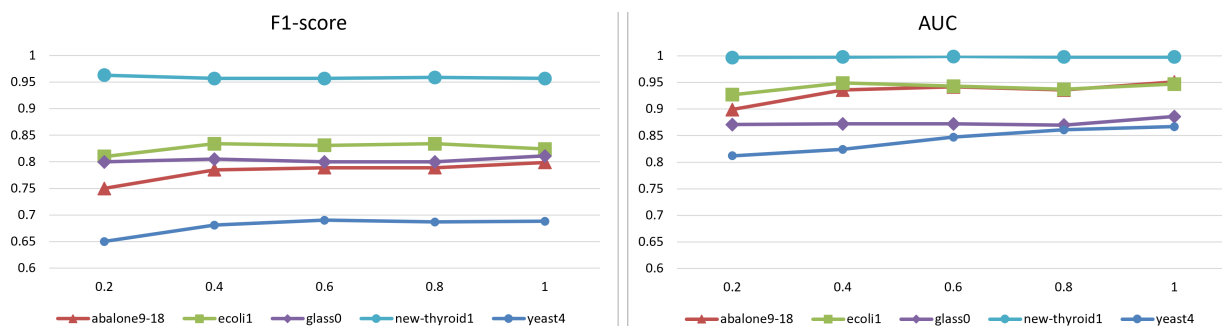


Figure 3.7: F1-score and AUC results with varying Gaussian standard deviation.

### 3.7.5 Empirical Study on the Impact of Informative Portion

This section studies the empirical impact of the informative portion (IP) in Section 3.2.3. This portion works as a threshold to adjust how many samples are taken into consideration of informative samples. To save space, we study five datasets used in Section 3.7.4. Different values of IP ranging from 0.1 to 1 are applied, and the classification performance results are shown in Figure 3.8.



As we observe from the figure, while the performance is not significantly affected to datasets achieved high performance (new-thyroid1, ecoli1), there are obvious peaks within the IP values of (0.2, 0.6) in both F1-score and AUC score for other datasets (abalone9-18, glass0, yeast4). This suggests that tuning IP for each dataset between a range of (0.2, 0.6) could achieve higher performance. For example, by adjusting IP from 0.1 to 0.3 in abalone9-18 dataset experiment, we can increase the performance by 5% for both F1-score and AUC score.

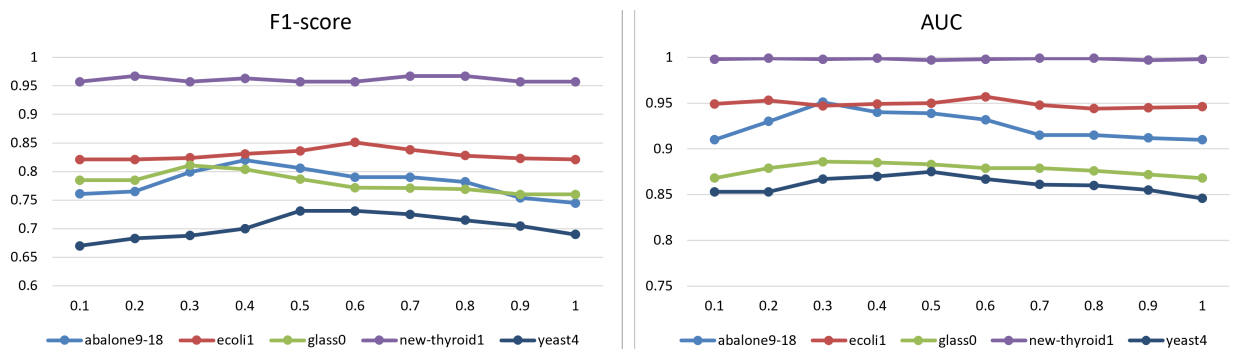


Figure 3.8: F1-score and AUC results with varying informative portion IP.

### 3.8 Conclusion

We propose a data balancing technique by generating synthetic data for minority samples maximizing the posterior ratio to embrace the chance they fall into the minority class and do not fall across the expected decision boundary. While maximizing the posterior ratio, we use kernel density estimation to estimate the likelihood so that it is able to work with complex distribution data without requiring data distribution assumptions. In addition, our technique leverage entropy-based active learning to find and balance the most informative samples. This is important to improve model performance as shown in our experiments on

41 real-world datasets. For future work, we would like to investigate the class imbalance for image data type and enhance our technique to adapt to image datasets.

## Chapter 4: AutoGAN-Based Dimension Reduction for Privacy Preservation <sup>1</sup>

### 4.1 Introduction

Machine Learning (ML) is an important aspect of modern applications that rely on big data analytics (e.g., an on-line system collecting data from multiple data owners). However, these applications are progressively raising many different privacy issues as they collect different types of data on a daily basis. For example, many types of data are being collected in smart cities such as patient records, salary information, biological characteristics, Internet access history, personal images and so on. These types of data then can be widely used in daily recommendation systems, business data analysis, or disease prediction systems which in turn affect the privacy of individuals who contributed their sensitive data. Considering a multi-level access control system of a company using biometric recognition (e.g., face recognition, fingerprint) for granting permission to access data resources, the company staff members may concern their biological information being vulnerable to adversaries. Even though the utility of these biometric features can be effectively used in machine learning tasks for authentication purpose, leaking this information might lead to privacy breaches. For example, an adversary could utilize them to determine the members' identities.

Several tools and methods have been developed to preserve the privacy in machine learning applications, such as homomorphic encryption [11, 26, 42], secure multi-party computing [100, 83], differential privacy (DP) [12, 103, 74, 6, 94], compressive privacy [108, 18, 17, 106, 52, 53, 95] and so on. Typically, differential privacy-based methods aim at preventing leaking individual information caused by queries. However, they are not designed to serve large number of queries since they require adding huge amount of noise to preserve

---

<sup>1</sup>This chapter was published in Journal of Neurocomputing. Permission is included in Appendix A.

privacy, thus significantly decreasing the ability to learn meaningful information from data. On the other hand, homomorphic encryption-based methods can be used to privately evaluate a function over encrypted data by a third party without accessing to plain-text data, hence the privacy of data owners can be protected. However, due to the high computational cost and time consumption, they may not work with a very large dataset, normally required in ML applications.

In this study, we consider an access control system collecting dimension-reduced face images of staff members to perform authentication task and to provide permission for members who would like to access company’s data resources (Figure 4.1). We propose a non-linear dimension reduction framework to decrease data dimension for the authentication purpose mentioned above and to protect against an adversary from reconstructing member images. Firstly, we introduce  $\epsilon$ -DR Privacy as a theoretical tool for dimension reduction privacy evaluation. It evaluates the reconstruction distance between original data and reconstructed data of a dimension reduction (DR) mechanism. This approach encourages a DR mechanism to enlarge the distance as high distance yields high level of privacy. While other methods such as differential privacy-based methods rely on inference uncertainty to protect sensitive data,  $\epsilon$ -DR Privacy is built on reconstruction error to evaluate privacy. Therefore, unlike differential privacy methods,  $\epsilon$ -DR Privacy is not negatively impacted by the number of queries. Secondly, as detailed in Section 4.3, we recommend a privacy-preserving framework Autoencoder Generative Adversarial Nets-based Dimension Reduction Privacy (AutoGAN-DRP) for enhancing data owner privacy and preserving data utility. The *utility* herein is evaluated via machine learning task performance (e.g., classification accuracy).

Our dimension reduction (DR) framework can be applied to different types of data and used in several practical applications without heavy computation of encryption and impact of query number. The proposed framework can be applied directly to the access control system mentioned above. More elaboratively, face images are locally collected, nonlinearly compressed to achieve DR, and sent to the authentication center. The server then performs

classification tasks on the dimension-reduced data. We assume the authentication server is semi-honest, that is to say it does not deviate from authenticating protocols while being curious about a specific member’s identity. Our DR framework is designed to resist against reconstruction attacks from a strong adversary who obtains the training dataset and the transformation model.

During the stage of experiments, we implemented our framework to evaluate dimension-reduced data in terms of accuracy of the classification tasks, and we attempted to reconstruct original images to examine the capacity of adversaries. We performed several experiments on three facial image datasets in both gray-scale and color, i.e., *the Extended Yale Face Database B* [33], *AT&T* [80], and *CelebFaces Attributes Dataset (CelebA)* [66]. The experiment results illustrate that with only seven reduced dimensions our method can achieve accuracies of 93%, 90%, and 80% for AT&T, YaleB, and CelebA respectively. Further, our experiments show that at the accuracies of 79%, 80% and 73% respectively, the reconstructed images could not be recognized by human eyes. In addition, the comparisons shown in Section 4.6 also illustrate that AutoGAN-DRP is more resilient to reconstruction attacks compared to related works. Our work has two main contributions:

1. To analytically support privacy guarantee, we introduce  $\epsilon$ -DR Privacy as a theoretical approach to evaluate privacy preserving mechanism.
2. We propose a non-linear dimension reduction framework for privacy preservation motivated by Generative Adversarial Nets [36] and Auto-encoder Nets [9].

The rest of this chapter is organized as follows. Section 4.2 summarizes state-of-the-art privacy preservation machine learning (PPML) techniques and reviews knowledge of deep learning methods including generative adversarial neural nets and Auto-encoder. Section 4.3 describes the privacy problem through a scenario of a facial recognition access control system, introduces the definition of  $\epsilon$ -DR Privacy to evaluate DR-based privacy preserving mechanisms, and presents our framework AutoGAN-based Dimension Reduction for Privacy

Preservation. Section 4.4 presents and discusses our experiment results over three different face image datasets. Section 4.5 compares AutoGAN-DRP to a similar work GAP in terms of reconstruction error and classification accuracy. Section 4.6 demonstrates reconstructed images over AutoGAN-DRP and other privacy preservation techniques (i.e., Differential Privacy and Principle Component Analysis). Finally, the conclusion and future work are mentioned in Section 4.7.

## 4.2 Related Work

### 4.2.1 Literature Review

Machine learning privacy preservation methods are categorized into two main approaches. Cryptographic approach applies to the scenarios where the data owners do not wish to expose their plain-text sensitive data while asking for machine learning services from a third-party. The most common tool used in this approach is fully homomorphic encryption that supports multiplication and addition operations over encrypted data, which enabling the ability to perform a more complex function. However, the high cost of the multiplicative homomorphic operations renders it difficult to be applied on machine learning tasks. In order to avoid multiplicative homomorphic operations, additive homomorphic encryption schemes are more widely used in privacy preserving machine learning (PPML). However, the limitation of the computational capacity in additive homomorphic schemes narrows the ability to apply on particular ML techniques. Thus, such additive homomorphic encryption-based methods in [11, 26, 10, 43] are only applicable to simple machine learning algorithms such as decision tree and naive bayes. In Hesamifard’s work [42], the fully homomorphic encryption is applied to perform deep neural networks over encrypted data, where the non-linear activation functions are approximated by polynomials.

In secure multi-party computing (SMC), multiple parties collaborate to compute functions without revealing plain-text to other parties. A widely-used tool in SMC is garbled circuit [100], a cryptographic protocol carefully designed for two-party computation, in which

they can jointly evaluate a function over their sensitive data without the trust of each other. In [8], Mohammad introduced a SMC protocol for principal component analysis (PCA) which is a hybrid system utilizing additive homomorphic and garbled circuit. In secret sharing techniques [83], a secret  $\mathbf{s}$  is distributed over multiple pieces  $\mathbf{n}$  also called *shares*, where the secret can only be recovered by a sufficient amount of  $\mathbf{t}$  *shares*. A good review of secret sharing-based techniques and encryption-based techniques for PPML is given in [73]. Although these encryption-based techniques can protect the privacy in particular scenarios, their computational cost is a significant concern. Furthermore, as [73] elaborated, the high communication cost also poses a big concern for both techniques.

The other category is Non-Cryptographic approach. For example, Differential Privacy (DP) [25] aims to prevent membership inference attacks. DP considers a scenario that an adversary infers a member’s information based on the difference of outputs of a ML mechanism before and after the member join a database. The database with the member’s information and without the member’s information can be considered as two neighbor databases which differ by at most one element. DP adds noise to the outputs of the ML mechanism to result in similar outputs from the two neighbor databases. Thus, adversaries cannot differentiate the difference between the two databases. A mechanism  $M$  satisfies  $\epsilon$ -differential privacy if for any two neighbor databases  $D$  and  $D'$ , and any subset  $S$  of the output space of  $M$  satisfies  $Pr[M(D) \in S] \leq e^\epsilon Pr[M(D') \in S]$ . The similarity of query outputs protects a member information from such membership inference attacks. The *similarity* is guaranteed by the parameter  $\epsilon$  in a mechanism in which the smaller  $\epsilon$  provides a better level of privacy preservation. [12, 103, 13, 93, 99] propose methods to guarantee  $\epsilon$ -differential privacy by adding noise to outcome of the weights  $\mathbf{w}^* = \mathbf{w} + \eta$ , where  $\eta$  drawn from Laplacian distribution and adding noise to the objective function of logistic regression or linear regression models. [74, 6] satisfy differential privacy by adding noise to the objective function while training a deep neural network using stochastic gradient descent as the optimization algorithm.

In addition, there are existing works proposing differential privacy dimension reduction. One can guarantee  $\epsilon$ -differential privacy by perturbing dimension reduction outcome. Principal component analysis (PCA) whose output is a set of eigenvectors is a popular method in dimension reduction. The original data is then represented by its projection on those eigenvectors, which keeps the largest variance of the data. One can reduce the data dimension by eliminating insignificant eigenvectors which contain less variance, and apply noise on the outcome to achieve differential privacy[94]. However, the downside of these methods is that they are designed for specific mechanisms and datasets and not working well with the others. For example, record-level differential privacy is not effectively used with image dataset as shown in [44]. Also, the amount of added noise is accumulative based on the number of queries so that this approach usually leads to low accuracy results with a high number of queries.

Similar to our work, Generative Adversarial Privacy (GAP) [18] is a perturbation method utilizing the minimax algorithm of Generative Adversarial Nets to preserve privacy and to keep utility of image datasets. GAP perturbs data within a specific  $l_2$  distance constraint between original and perturbed data to distort private class labels and at the same time preserve non-private class labels. However, it does not protect the images themselves, and an adversary can visually infer private label (e.g., identity) from images. In contrast, our method protects an image by compressing it into a few dimension vector and then transferring without clearly exposing the original image.

#### 4.2.2 Preliminaries

To enhance the distance between original and reconstructed data in our DR system, we utilize the structure of Generative Adversarial Network (GAN) [36] for data perturbation and deep Auto-encoder [9] for data reconstruction. The following sections briefly review Auto-encoder and GAN.



#### 4.2.2.1 Auto-encoder

Auto-encoder is aimed at learning lower dimension representations of unsupervised data. Auto-encoder can be used for denoising and reducing data dimension. It can be implemented by two neural network components: *encoder* and *decoder*. The *encoder* and *decoder* perform reverse operations. The input of the *encoder* is the original data while the output of the *decoder* is expected to be similar to the input data. The middle layer extracts latent representation of original data that could be used for dimension reduction. An Auto-encoder training process can be described as a minimization problem of the auto-encoder's loss function  $\mathcal{L}(\cdot)$ :

$$\mathcal{L}(x, g(f(x))) \quad (4.1)$$

where  $x$  is input data,  $f(\cdot)$  is an encoding function, and  $g(\cdot)$  is a decoding function.

#### 4.2.2.2 GAN

Generative Adversarial Nets is aimed at approximating distribution  $p_d$  of a dataset via a generative model. GAN simultaneously trains two components *generator*  $G$  and *discriminator*  $D$ , and the input of  $G$  is sampled from a prior distribution  $p_z(z)$  through which  $G$  generates fake samples similar to the real samples. At the same time,  $D$  is trained to differentiate between fake samples and real samples, and send feedback to  $G$  for improvement. GAN can be formed as a two-player minimax game with value function  $V(G,D)$ :

$$\min_G \max_D V(G, D) = E_{x \sim p_d} [\log(D(x))] + E_{z \sim p_z} [\log(1 - D(G(z)))] \quad (4.2)$$

The two components, *Generator* and *Discriminator* can be built from neural networks (e.g., fully connected neural network, convolutional neural network). The goal of  $G$  is to reduce the accuracy of  $D$ . Meanwhile, the goal of  $D$  is to differentiate fake samples from

real samples. These two components are trained until the discriminator cannot distinguish between generated samples and real samples.

### 4.3 Methodology

In this section, we first describe the problem and threat model, then we introduce a definition of DR-Privacy and our dimensionality reduction method (AutoGAN-DRP).

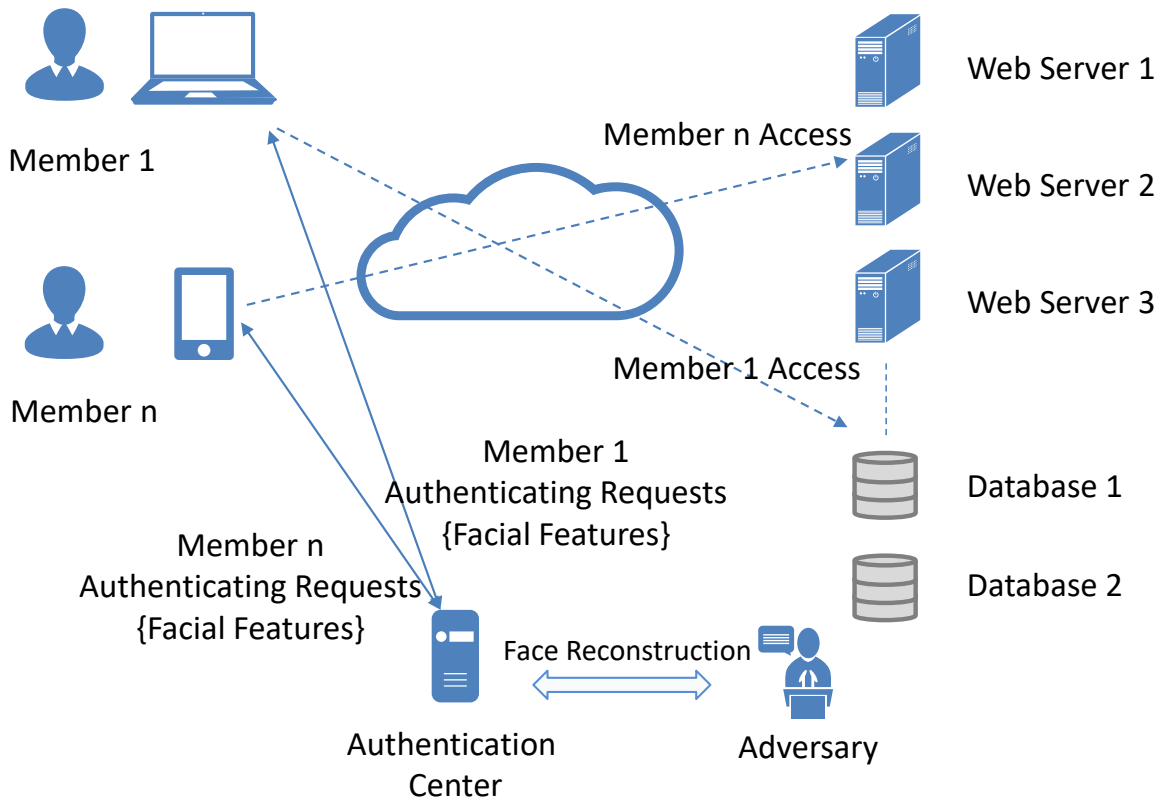


Figure 4.1: Attack model.

### 4.3.1 Problem Statement

We introduce the problem through the practical scenario mentioned in Section 4.1. Figure 4.1 briefly describes the entire system in which staff members (clients) in a company request access to company resources, such as websites and data servers through a face recognition access control system. For example, if member  $n$  requests to access web server 2, the local device first takes a facial photo of the member by an attached camera, locally transforms it into lower dimension data, and sends to an authentication center. The authentication server then obtains the low dimensional data and determines member access eligibility by using a classifier without clear face images of the requesting member. We consider that the system has three levels of privileges (i.e., single level, four-level, eight-level) corresponding to three groups of members. We assume the authentication server is semi-honest (it obeys work procedure but might be used to infer personal information). If the server is compromised, an adversary in the authentication center can reconstruct the face features to achieve plain-text face images and determine members' identity.

### 4.3.2 Threat Model

In the above scenario, we consider that a strong adversary who has access to the model and training dataset attempts to reconstruct the original face images for inferring a specific member's identity. Our attack model can be represented in Figure 4.1. The adversary utilizes training data and facial features to identify a member identity by reconstructing the original face images using a reconstructor in an auto-encoder. Rather than using fully connected neural network, we implement the auto-encoder by convolutional neural network which more effective for image datasets. Our goal is to design a data dimension reduction method for reducing data dimension and resisting full reconstruction of original data.

### 4.3.3 $\epsilon$ -Dimension Reduction Privacy ( $\epsilon$ -DR Privacy)

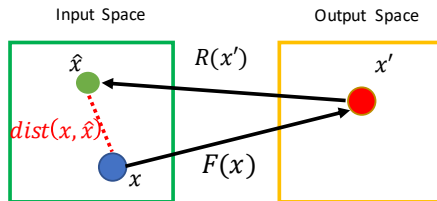


Figure 4.2: DR projection and reconstruction.

We introduce the Dimension Reduction Privacy (DR-Privacy), and define a formal definition of the  $\epsilon$ -DR Privacy to mathematically quantify/evaluate the mechanisms designed to preserve the DR-Privacy via dimension reduction. The DR-Privacy aims to achieve privacy-preserving via dimension reduction, which refers to transforming the data into a lower dimensional subspace, such that the private information is concealed while the underlying probabilistic characteristics are preserved, which can be utilized for machine learning purposes. To quantify the DR-Privacy and guide us to design such DR functions, we define  $\epsilon$ -DR Privacy as follows.

**Definition 1:** ( $\epsilon$ -DR Privacy) A Dimension Reduction Function  $F(\cdot)$  satisfies  $\epsilon$ -DR Privacy if for each i.i.d.  $m$ -dimension input sample  $x$  drawn from the same distribution  $D$ , and for a certain distance measure  $dist(\cdot)$ , we have

$$\mathbb{E}[dist(x, \hat{x})] \geq \epsilon \tag{4.3}$$

where  $\mathbb{E}[\cdot]$  is the expectation,  $\epsilon \geq 0$ ,  $x' = F(x)$ ,  $\hat{x} = R(x')$ , and  $R(\cdot)$  is the Reconstruction Function.

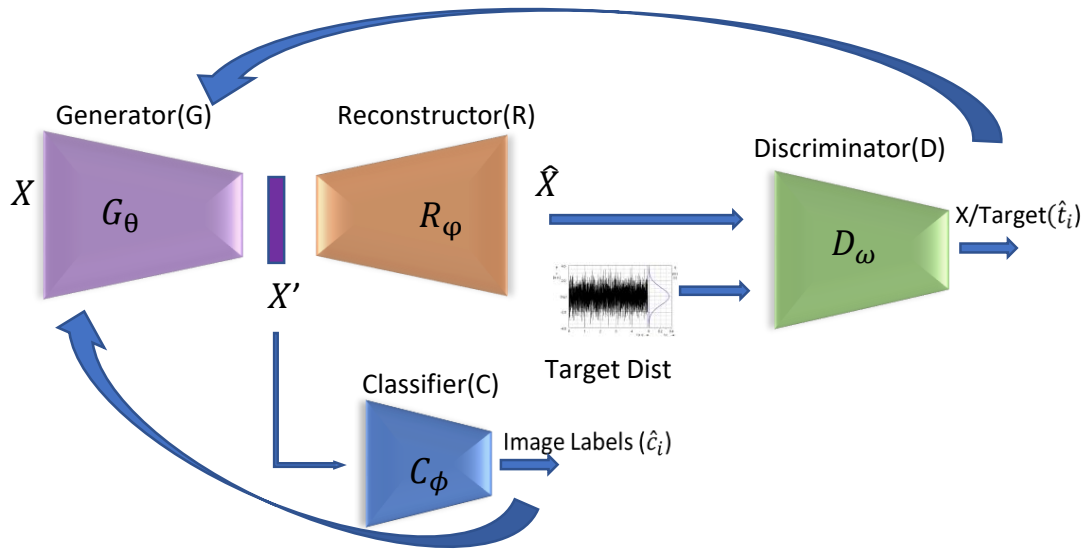


Figure 4.3: AutoGAN-DRP

For instance, as shown in Fig. 4.2, given original data  $x$ , our framework utilizes certain dimension reduction function  $F(x)$  to transform the original data  $x$  into the transformed data  $x'$ . The adversaries aim to design a corresponding reconstruction function  $R(x')$  such that the reconstructed data  $\hat{x}$  would be closed/similar to the original data  $x$ . DR-Privacy aims to design/develop such dimension reduction functions, that the distance between the original data and its reconstructed data would be large enough to protect the privacy of the data owner.

#### 4.3.4 AutoGAN Dimension Reduction for Privacy Preserving

We propose a deep learning framework for transforming face images to low dimensional data which is hard to be fully reconstructed. The framework can be presented in Figure 4.3. We leverage the structure of an auto-encoder [9] which contains encoder and decoder (in this work, we called them generator and re-structor) in order to reduce data dimension. More specifically, the low dimensional representations are extracted from the middle layer

of the auto-encoder (the output of the generator). The dimension-reduced data can be sent to the authentication server as an authentication request. We consider an adversary as a re-constructor implemented by a decoder. To resist against fully reconstructing images, the framework utilizes a discriminator in GAN [36] to direct reconstructed data to a designated target distribution with an assumption that the target distribution is different from our data distribution. In this work, the target distribution is sampled from Gaussian distribution and the mean is the average of training data. After projecting data into a lower dimension domain, the re-constructor is only able to partially reconstruct the data. Therefore, the adversary might not be able to recognize an individual’s identity. To maintain data utility, we also use feedback from a classifier. The entire framework is designed to enlarge the distance between original data and its reconstruction to preserve individual privacy and retain significant data information. The dimension-reduced transformation model is extracted from the framework and provided to clients for reducing their face image dimensions. The classification model will be used in an authentication center that classifies whether a member’s request is valid to have access (1) or not (0).

We formulate the problem as follows: Let  $X$  be the public training dataset.  $(x_i, y_i)$  is the  $i$ th sample in the dataset in which each sample  $x_i$  has  $d$  features and a ground truth label  $y_i$ . The system is aimed at learning a dimension reduction transformation  $F(\cdot)$  which transforms the data from  $d$  dimensions to  $d'$  dimensions in which  $d' \ll d$ . Let  $X'$  be the dataset in lower dimension domain. The dimension-reduced data should keep significant information to work with different types of machine learning tasks and should resist against the reconstruction or inference from data owner information.

Our proposed framework is designed to learn a DR function  $F(\cdot)$  that projects data onto low dimension space and preserves privacy at certain value of  $\epsilon$ . The larger distance implies higher level of privacy. Figure 4.3 presents our learning system in which the dimension-reduced data  $X'$  is given by a generator  $G$ . Since  $X'$  is expected to be accurately classified by a classifier  $C$ , the generator improves by receiving feedback from the classifier via the

classifier’s loss function  $\mathcal{L}_C$ . We use a binary classifier for single-level authentication system and multi-class classifiers for multi-level authentication system. The classifier loss function is defined as the cross entropy loss of the ground truth label  $y$  and predicted label  $\hat{y}$  as follows.

$$\mathcal{L}_C = - \sum_{i=1}^n \sum_{j=1}^m y_{ij} \log(\hat{y}_{ij}) \quad (4.4)$$

where  $m$  denotes the number of classes and  $n$  denotes the number of samples.

To evaluate data reconstruction and enlarge the reconstruction distance, a re-constructor  $R$  is trained as a decoder in an auto-encoder and sends feedback to the generator via its loss function  $\mathcal{L}_R$ . The re-constructor plays its role as an aggressive adversary attempting to reconstruct original data by using known data. The loss function of  $R$  is the mean square error of original training data ( $x$ ) and reconstructed data ( $\hat{x}$ ), as displayed in (4.5) as follows

$$\mathcal{L}_R = \sum_{i=1}^n (x_i - \hat{x}_i)^2 \quad (4.5)$$

To direct the reconstructed data to a direction that reveals less visual information, the generator is trained with a discriminator  $D$  as a minimax game in GAN. The motivation is to direct reconstructed data to a certain target distribution (e.g., normal distribution). To ensure a distance, the target distribution should be different to training data distribution. The discriminator aims to differentiate the reconstructed data from samples of the target distribution. The loss function of  $D$  ( $\mathcal{L}_D$ ) can be defined as a cross-entropy loss of ground truth labels (0 or 1)  $t$  and prediction labels  $\hat{t}$  shown in (4.6).

$$\mathcal{L}_D = - \sum_{i=1}^n (t_i \log(\hat{t}_i) + (1 - t_i) \log(1 - \hat{t}_i)) \quad (4.6)$$

The optimal generator parameter  $\theta^*$  is given by the optimization problem of the generator loss function  $\mathcal{L}_G$ :

$$\underset{\theta}{\text{minimize}} \mathcal{L}_G(\theta) = \alpha \underset{\phi}{\text{min}} \mathcal{L}_C - \beta \underset{\omega}{\text{min}} \mathcal{L}_D - \gamma \underset{\varphi}{\text{min}} \mathcal{L}_R + \mathcal{C}(\epsilon) \quad (4.7)$$

where  $\theta$ ,  $\phi$ ,  $\omega$ , and  $\varphi$  are the model parameters of the generator, classifier, discriminator, and re-constructor respectively.  $\alpha$ ,  $\beta$ , and  $\gamma$  are weights of components in the objective function of the generator and can be freely tuned.  $\mathcal{C}(\epsilon)$  is a constraint function with respect to hyper-parameter  $\epsilon$ , as to be elaborated in the following subsection.

#### 4.3.5 Optimization with Constraint

In order to meet a certain level of reconstruction distance, we consider the constrained problem:

$$\begin{aligned} & \underset{\theta}{\text{minimize}} \mathcal{L}_G(\theta) \\ & \text{s.t. } \mathbb{E}_{x \sim p_d}[\text{dist}(x, \hat{x})] \leq \epsilon \end{aligned} \quad (4.8)$$

The optimization problem above can be approximated as an unconstrained problem [48]:

$$\underset{\theta}{\text{minimize}} (\mathcal{L}_G(\theta) + \gamma \mathcal{C}(\epsilon)) \quad (4.9)$$

where  $\gamma$  is a penalty parameter and  $\mathcal{C}$  is a penalty function

$$\mathcal{C}(\epsilon) = \max(0, \mathbb{E}_{x \sim p_d}[\text{dist}(x, \hat{x})] - \epsilon) \quad (4.10)$$

Note that  $\mathcal{C}$  is nonnegative, and  $\mathcal{C}(\theta) = 0$  iff the constraint in (4.8) is satisfied.

#### 4.3.6 Training Algorithms



---

**Algorithm 4.1** Algorithm for stochastic gradient descent training of  $\epsilon$ -DR Privacy.

---

**Input:** Training dataset  $X$ .

Parameter: learning rate  $\alpha_r, \alpha_d, \alpha_c, \alpha_g$ , training steps  $n_r, n_d, n_c, n_g$

A constraint for  $\epsilon$ -DR

**Output:** Transformation Model

*Initialization.*

```
1: for  $n$  global training iterations do
2:   Randomly sample a mini batch from target distribution and label  $\mathbf{t}$ .
3:   Randomly sample mini batch of data  $\mathbf{x}$  and corresponding label  $\mathbf{y}$ 
4:   for  $i = 0$  to  $n_r$  iterations do
5:     Update the Reconstruction:
6:      $\varphi_{i+1} = \varphi_i - \alpha_r \nabla_{\varphi} \mathcal{L}_R(\varphi_i, \mathbf{x})$ 
7:   end for
8:   for  $j = 0$  to  $n_d$  iterations do
9:     Update the Discriminator parameter:
10:     $\omega_{j+1} = \omega_j - \alpha_d \nabla_{\omega} \mathcal{L}_D(\omega_j, \mathbf{x}, \mathbf{t})$ 
11:   end for
12:   for  $k = 0$  to  $n_c$  iterations do
13:     Update the Classifier parameter:
14:      $\phi_{k+1} = \phi_k - \alpha_c \nabla_{\phi} \mathcal{L}_C(\phi_k, \mathbf{x}, \mathbf{y})$ 
15:   end for
16:   for  $l = 0$  to  $n_g$  iterations do
17:     Update the Generator parameter:
18:      $\theta_{l+1} = \theta_l - \alpha_g \nabla_{\theta} \mathcal{L}_G(\theta_l, \mathbf{x}, \mathbf{t}, \mathbf{y})$ 
19:   end for
20: end for
21: return
```

---

Algorithm 4.1 describes the training process of AutoGAN-DRP. The framework contains four components, and they are trained one by one (lines 4-15) within one global training step. After sampling batches from target distribution and data for inputs of the models (lines 2-3), we then train the four components. First, the re-constructor is trained in  $n_r$  iterations while other components' parameters are fixed (lines 4-6). Second, the discriminator is trained (lines 7-9). Third, the classifier is trained in  $n_c$  iterations (lines 10-12). Fourth, the generator is trained in  $n_g$  iterations (lines 13-15). After training each component in their number of local training steps, the above training process is repeated until it reaches the number of

global training iterations (lines 1-16). In our setting, the numbers of local training iterations ( $n_c, n_r, n_d, n_g$ ) are much smaller than the number of global iterations  $n$ .

#### 4.4 Experiments and Discussion

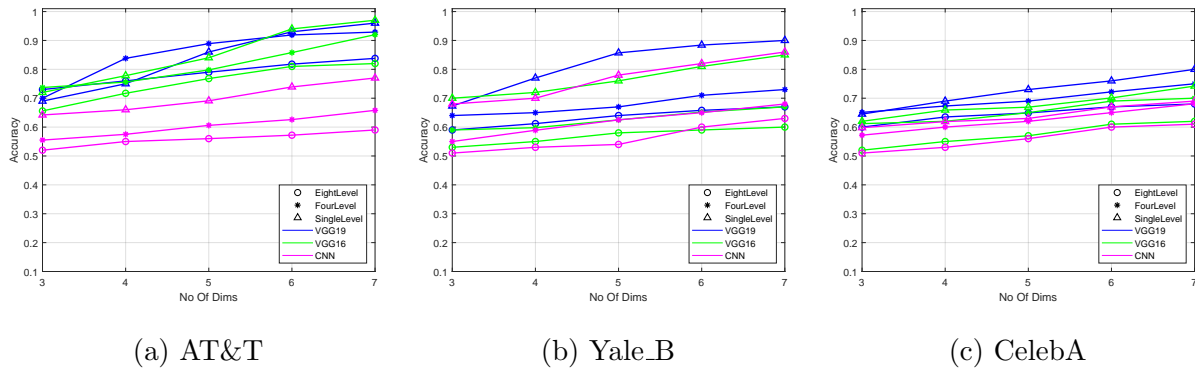


Figure 4.4: Accuracy for different number of reduced dimensions.

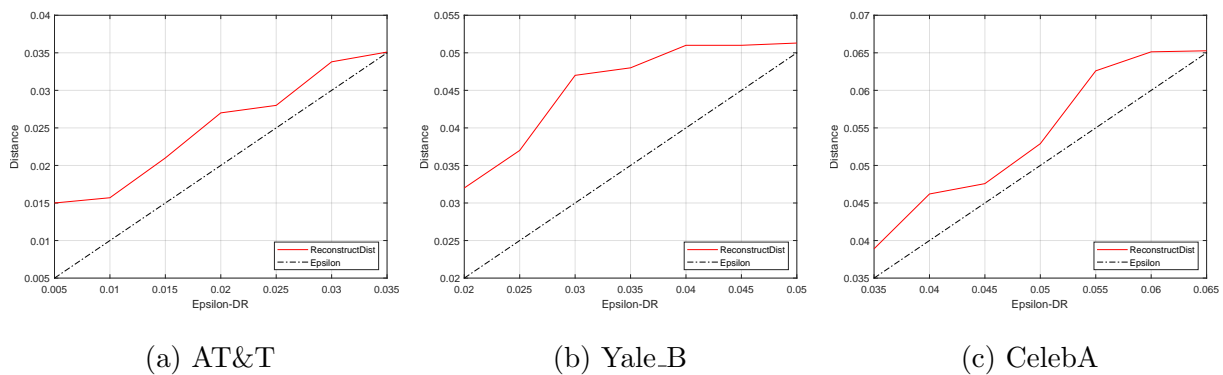


Figure 4.5: Average distance measurement result.

Table 4.1: Implementation information

	VGG16			VGG19			CNN		
	Hidden layers	Units	Parameter	Hidden layers	Units	Parameter	Hidden layers	Units	Parameter
Generator	Conv_block Max_pooling Conv_block Max_pooling Conv_block Max_pooling Conv_block Max_pooling Conv_block Max_pooling Dense Dense	64×2 128×2 256×3 512×3 512×3 1024 1024	16,295,623	Conv_block Max_pooling Conv_block Max_pooling Conv_block Max_pooling Conv_block Max_pooling Conv_block Max_pooling Dense Dense	64×2 128×2 256×4 512×4 512×4 1024 1024	21,605,319	Conv BatchNorm Conv BatchNorm Conv BatchNorm Dense	256 512 1024 1024	16,451,847
Reconstructor	Dense Dense Dense Reshape Conv_block-T Up_sampling Conv_block-T Up_sampling Conv_block-T Up_sampling Conv_block-T Up_sampling Conv_block-T Up_sampling Conv_block-T	1024 1024 1024 512×3 512×3 256×3 128×2 64×2	10,184,000	Dense Dense Dense Reshape Conv_block-T Up_sampling Conv_block-T Up_sampling Conv_block-T Up_sampling Conv_block-T Up_sampling Conv_block-T Up_sampling Conv_block-T	1024 1024 1024 512×4 512×4 256×4 128×2 64×2	13,281,472	Dense BatchNorm Reshape Conv BatchNorm Conv BatchNorm Conv	1024 1024 1024 512 256	18,048,256
Classifier	Dense BatchNorm Dropout Dense BatchNorm Dropout Dense BatchNorm Dropout Dense BatchNorm Dropout	2048 2048 2048 2048	12,636,168	Dense BatchNorm Dropout Dense BatchNorm Dropout Dense BatchNorm Dropout Dense BatchNorm Dropout	2048 2048 2048 2048	12,636,168	Dense BatchNorm Dropout Dense BatchNorm Dropout Dense BatchNorm Dropout Dense BatchNorm Dropout	2048 2048 2048 2048	12,636,168
Discriminator	Conv Dropout Conv Dropout Flatten Dense Dense	128 256 1024 1024	5,084,737	Conv Dropout Conv Dropout Flatten Dense Dense	128 256 1024 1024	5,084,737	Conv Dropout Conv Dropout Flatten Dense Dense	128 256 1024 1024	5,084,737
Shared parameters: optimizer Adam, learning rate 0.0001, 7 dimensions Hardware: GPU Testla T4 16Gb, CPU Xeon Processors @2.3Ghz Software: Tensorflow 2.0 beta. The number of trainable parameters are reported by model.summary() from Keras library.									

In this section, we demonstrate our experiments over three popular supervised face image datasets: *the Extended Yale Face Database B* [33], *AT&T* [80], and *CelebFaces Attributes Dataset (CelebA)* [66]. To comprehensively evaluate our method performance, we also conduct experiments with different generator and re-constructor structures, different types of classifications (binary and multi-class classification), different numbers of reduced dimensions. The effectiveness of the method is then evaluated in terms of utility and privacy.

#### 4.4.1 Experiment Setup

*The Extended Yale Face Database B* (YaleB) contains 2,470 grayscale images of 38 human subjects under different illumination conditions and their identity label. In this dataset, the image size is  $168 \times 192$  pixels. The AT&T dataset has 400 face images of 40 subjects. For convenience, we resize each image of these two dataset to  $64 \times 64$  pixels. CelebA is a color facial image dataset containing 202,599 images of 10,177 subjects. 1,709 images of the first 80 subjects are used for our experiment. Each image is resized to  $64 \times 64 \times 3$  pixels. All pixel values are scaled to the range of  $[0,1]$ . We randomly select 10% of each subject’s images for validation and 15% for testing dataset.

The generator and re-constructor in Figure 4.3 are implemented by three different structures. Specifically, we follow the architecture of recent powerful models VGG19, VGG16 [87] and a basic convolutional network (CNN). We modify the models to adapt to our data size ( $64 \times 64$ ). Discriminator and Classifier are built on fully connected neural network and convolutional network respectively. Leaky ReLU is used for activation function in hidden layers. We use linear activation function for generator’s output layers and softmax activation functions for other components’ output layers. Each component is trained in 5 local iterations ( $n_r, n_g, n_d, n_c$ ), and the entire system is trained in 500 global iterations ( $n$ ). The target distribution is drawn from Gaussian distribution (with the covariance value of 0.5 and the mean is the average of the training data). Table 4.1 provides detail information of neural networks’ structures and other implementation information.

To evaluate the reliability, we test our framework with different levels of authentication corresponding to binary classification (single-level) and multi-class classification (multi-level). For the single-level authentication system, we consider half of the subjects in the dataset are valid to access company’s resources while the rest are invalid. We randomly divide the dataset into two groups of subjects and labels their images to (1) or (0) depending on their access permission. For the cases of multi-level authentication system, we divide the subjects into four groups and eight groups. Therefore, the authentication server becomes four-class and eight-class classifier respectively.

#### 4.4.2 Utility

We use accuracy metric to evaluate the utility of dimension-reduced data. The testing dataset is tested with the classifier extracted from our framework. Different structures of Generator and re-constructor are applied including VGG19, VGG16, basic CNN on different privilege levels which correspond to multi-class classification. Figure 4.4 illustrates the accuracies for different dimensions from three to seven over the three facial datasets. Overall, the accuracies improve when the number of dimension increases. The accuracies on the two gray image datasets (AT&T and Yale\_B) reaches 90% and higher when using VGG with only seven dimensions. This accuracy figure for Celeba is smaller, but it still reaches 80%. In general, VGG19 structure performs better than using VGG16 and basic CNN in terms of utility due to the complexity (table 4.1) and adaptability to image datasets of VGG19. As the dimension number is reduced from 4,096 ( $64 \times 64$ ) to 7, we can achieve a compression ratio of 585 yet achieve accuracy of 90% for the two gray datasets and 80% for the color dataset. This implies our method could gain a high compression ratio and maintain a high utility in terms of accuracy. During conducting experiments we also observe that the accuracy could be higher if we keep the original resolution of images. However, for convenience and reducing the complexity of our structure, we resize images to the size of  $64 \times 64$  pixels.

### 4.4.3 Privacy

In this study, the Euclidean distance is used to measure the distance between original and reconstructed images:  $dist(x, \hat{x}) = \|x - \hat{x}\|^2$ . Figure 4.5 illustrates the average distances between original images and reconstructed images on testing data with different  $\epsilon$  constraints (other setting parameters: seven dimensions, single-level authentication, and VGG19 structure). The achieved distances (red lines) are larger than the hyper-parameter  $\epsilon$  (black dotted lines) where  $\epsilon$  is less than 0.035 for AT&T, 0.052 for YaleB and 0.067 for CelebA. Thus, our framework can satisfy  $\epsilon$ -DR with  $\epsilon$  of above values. Due to the fact that the re-constructor obtained some information (we consider the adversary can reach the model and the training data), we can only set the distance constraint  $\epsilon$  within a certain range as shown in 4.5. The intersection between the red line and the dotted black line points out the largest distance our framework can achieve. Since the mean of the target distribution is set to be the same as the mean of training dataset, reconstructed images will be close to the mean of training dataset which we believe it will enlarge the distance and expose less individual information. Thus, the range of epsilon can be estimated base on the expectation of the distance between testing samples and the mean of training data. In addition, the first section of Table 4.2 demonstrates some samples and their corresponding reconstructions in single-level authentication and seven dimensions with different achieved accuracies and distances. The reconstructed images could be nearly identical, thus making it visually difficult to recognize the identity of an individual.

## 4.5 Comparison to GAP[18]

In this section, we compare the proposed framework with GAP, which shares many similarities. At first, we attempt to visualize AutoGAN-DRP and GAP by highlighting their similarities and differences. Then, we exhibit our experiment results of the two methods on the same dataset.

In terms of similarities, AutoGAN-DRP and GAP are utilizing minimax algorithms of Generative Adversarial Nets, applying the state-of-the-art convolution neural nets for image datasets, considering  $l_2$  norm distance (i.e., distortion in GAP, privacy measurement in AutoGAN-DRP) between the original images and reconstructed images. Specifically, both GAP and AutoGAN-DRP consider the reconstruction distance between original and reconstructed images. In GAP this *distortion* refers to the Euclidean between original and privatized images, and AutoGAN-DRP denotes the *distance* as the Euclidean distance between original and reconstructed images. In this context, the distance and distortion refer to the same measurement and have the same meaning. To be consistent, we use the term *distance* to present this measurement in the rest of this section.

However, there are also distinctions between GAP and AutoGAN-DRP. In GAP, the adversary aims to identify a private label (e.g., gender) which should be kept secret while AutoGAN-DRP aims to visually protect the owner’s face images by enlarging the reconstruction distance. Thus, instead of considering a private label in loss function of the generator in GAP, AutoGAN-DRP is aimed at driving the reconstructed data into a target distribution using a discriminator.

Figure 4.6 illustrates the visualization of AutoGAN-DRP and GAP. In AutoGAN-DRP, privacy is assessed based on how well an adversary can reconstruct the original data and measured by the distance between original and reconstructed data. The dimension-reduced data is reconstructed using the state-of-the-art neural network (an Auto-encoder). The larger the distance is, the more privacy can be achieved. Further, if the reconstructed images are blurry, privacy can be preserved since it is hard to visually determine an individual identity. The data utility is quantified by the accuracy of the classification tasks over dimension-reduced data which captures the most significant data information. Meanwhile, GAP perturbs images with a certain distortion constraint to achieve privacy. It evaluates data utility by the classification accuracy of non-private label and assesses privacy by the classification accuracy of private label. Similar to AutoGAN-DRP, the high distortion is most likely to yield high

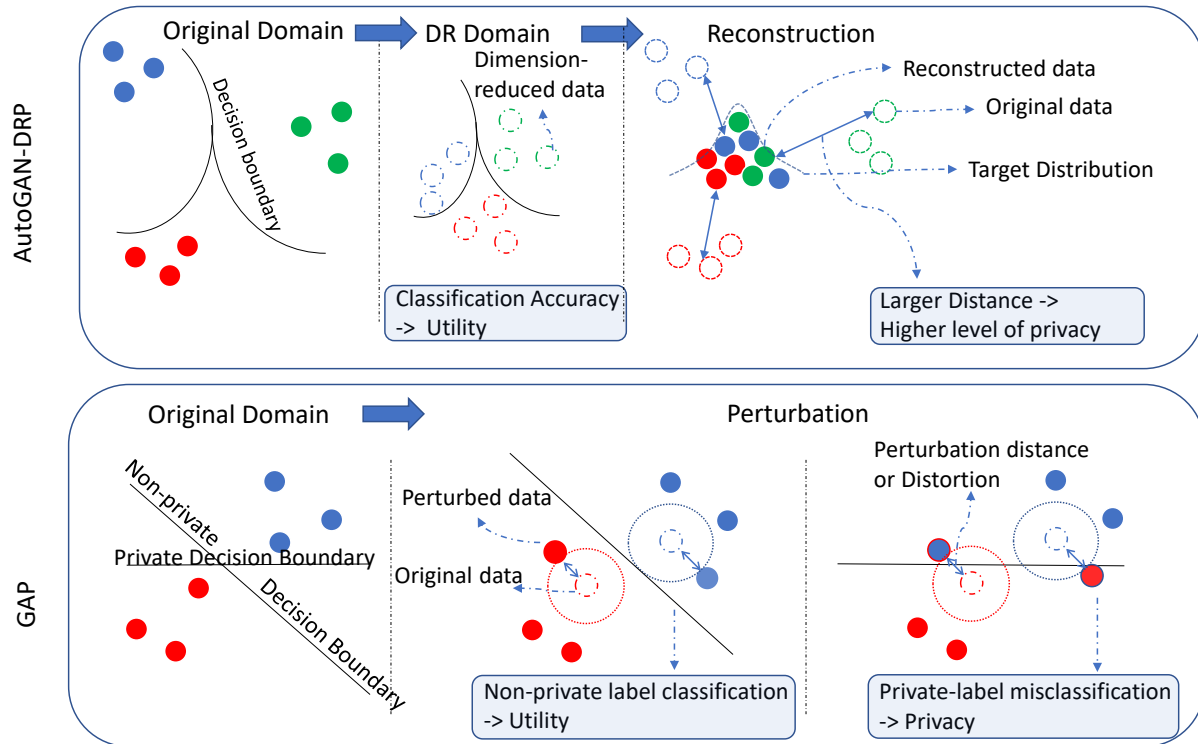


Figure 4.6: AutoGAN-DRP and GAP visualization.

level of privacy. In GAP, however, high distortion might dramatically reduce the classification accuracy of non-private label. This might be caused by the high correlation between private and non-private labels. This difference enables AutoGAN-DRP to preserve more utility than GAP at the same distortion level, as the experiment result (depicted in Figure 4.7) reveals.



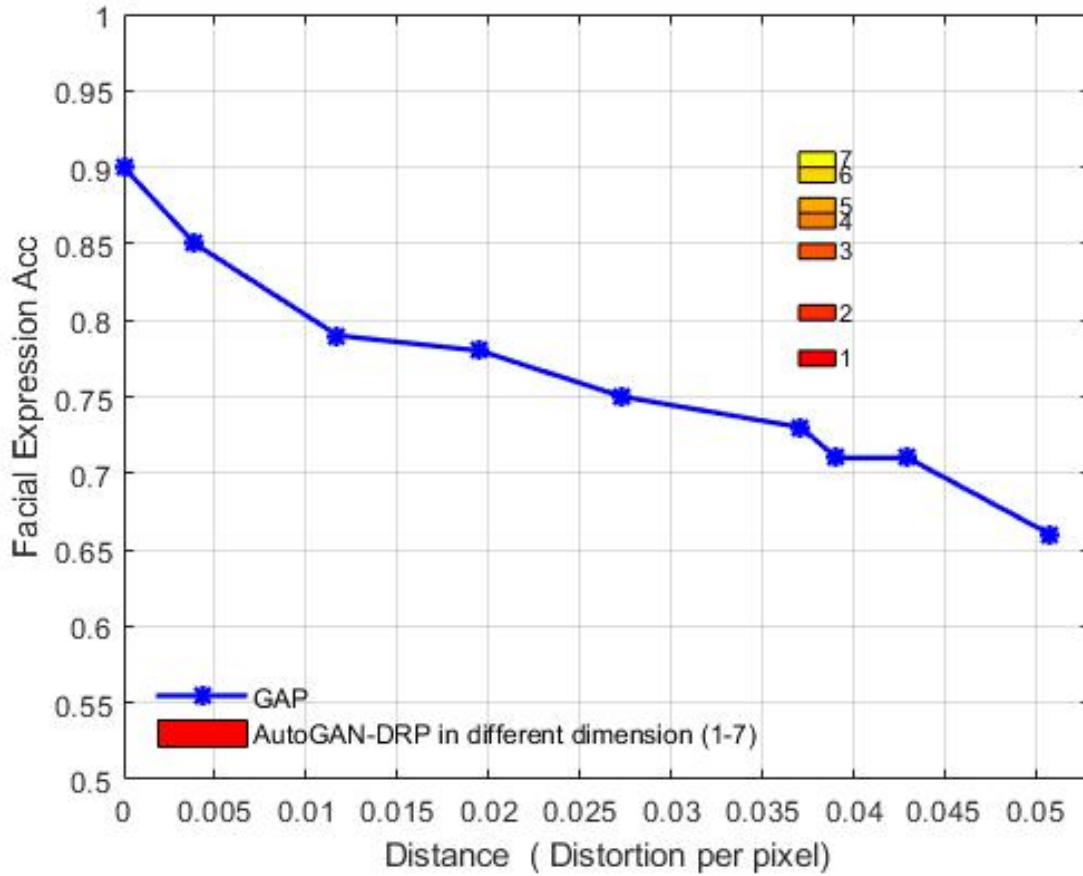


Figure 4.7: GENKI Facial Expression Vs AutoGAN-DRP Distance.

In the experiment, we reproduce a prototype of Transposed Convolutional Neural Nets Privatizer (TCNNP) in GAP using materials and source code provided by [18]. We also modify our framework to make it as similar to TCNNP as possible. Specifically, a combination of two convolutional layers with ReLU activation function and two fully connected neural network layers are used for implementing the Generator similar to TCNNP. Our Classifier is constructed on two convolutional layers and two fully connected hidden layers similar to the Adversary in GAP. We also test our framework on GENKI, the same dataset with GAP. The utility is evaluated by the accuracy of facial expression classification (a binary classification). It should be noted that our framework have been shown to work on different datasets with

multi-class classification, which is more challenging and comprehensive. Figure 4.7 shows the accuracy results of GAP and AutoGAN-DRP for GENKI dataset. AutoGAN-DRP achieves distances ranging from 0.037 to 0.039 for different dimensions from one to seven. At the same range of distance (distortion per pixel), GAP achieves accuracy of only 72% while AutoGAN-DRP gains accuracy rates starting from 77% to 91% for different number of dimensions. It becomes evident that our method can achieve higher accuracy than that of GAP at the same distortion level.



#### 4.6 Visual Comparison to Privacy Preserving Techniques

In this section, we compare AutoGAN-DRP with other privacy preserving methods in terms of ability to visually identify client’s identities. We choose the widely used tool for privacy preserving Differential Privacy (DP) [25] and another privacy preservation method utilizing dimensionality reduction technique (i.e., Principle Component Analysis [31] ).

In these experiments, we implement AutoGAN-DRP following VGG19 structure for the Generator and Re-constructor, and other setting parameters (e.g., number of hidden layers, learning rate, optimization) are shown in Table 4.1. The images are reduced to seven dimensions for different values of  $\epsilon$ -DR to achieve different distances and accuracies. The datasets are grouped into two groups corresponding to a binary classifier.

For implementing DP, we first generate a classifier on the authentication server by training the datasets with a VGG19 binary classifier (the structure of hidden layers is similar to our Generator in Table 4.1). The testing images are then perturbed using differential privacy method. Specifically, Laplace noise is added to the images with the sensitivity coefficient of 1 (it is computed by the maximum range value of each pixel [0,1]) and different DP epsilon parameters (this DP epsilon is different from our  $\epsilon$ -DR). The perturbed images are then sent to the authentication server and fed to the classifier. We visually compare the perturbed images of this method with AutoGAN.

Table 4.2: Sample visualization of AutoGAN, DP, PCA over three datasets

		AT&T			YaleB			CelebA				
Acc	Dist	0.93	0.79	0.65	0.90	0.80	0.69	0.73	0.66	0.59		
Dist		0.0116	0.0198	0.0245	0.0184	0.0246	0.0585	0.0513	0.0531	0.06618		
AutoGAN-DRP												
												
												
												
	Org	(7)	(7)	(7)	Org	(7)	(7)	(7)	Org	(7)	(7)	(7)
Acc	Dist	0.69	0.63	0.57	0.68	0.60	0.58	0.62	0.59	0.56		
Dist		0.0164	0.0313	0.0405	0.0149	0.0314	0.0407	0.0200	0.0418	0.0509		
Differential Privacy												
												
												
												
	Org	(11)	(8)	(7)	Org	(11)	(8)	(7)	Org	(11)	(8)	(7)
Acc	Dist	0.90	0.75	0.60	0.87	0.83	0.71	0.71	0.68	0.57		
Dist		0.0197	0.0264	0.0348	0.0228	0.0266	0.0287	0.0362	0.0379	0.0511		
PCA												
												
												
												
	Org	(15)	(7)	(5)	Org	(15)	(7)	(5)	Org	(15)	(7)	(5)

Acc : Average accuracy on testing data  
Dist: Average Euclidean distance between original images and reconstructed/perturbed images  
Org : Original images  
(.) Experiment parameters: epsilon for DP and number of reduced dimensions for PCA and AutoGAN-DRP

In addition, we follow instruction in FRiPAL [108] in which the clients reduce image dimension using Principle Component Analysis (PCA) and send reduced features to the server. FRiPAL claims that by reducing image dimension, their method can be more resilient to reconstruction attacks. The experiments are conducted with different number of reduced dimension. The images are reconstructed using *Moore–Penrose inverse* method with assumption that an adversary has access to the model. The classification accuracy is evaluated using a classifier which has similar structure to AutoGAN’s classifier.

Table 4.2 shows image samples and results over the three datasets. Overall, AutoGAN-DRP is more resilient to reconstruction attacks compared to the other two techniques. For instance, at the accuracy of 79% on AT&T dataset, 80% on YaleB, and 73% on CelebA, we cannot distinguish entities from the others. For DP method, the accuracy decreases when the DP epsilon decreases (adding more noise), and the perturbed images become harder to recognize. However, at a low accuracy 57%, we are still able to distinguish identities by human eyes. The reason is that DP noise does not focus on the important visual pixels. For PCA, the accuracy also goes down when the number of dimensions decreases and the distances increase. Since PCA transformation is linear and deterministic, the original information can be significantly reconstructed using the inverse transformation deriving from the model or training data. Thus, at the accuracy of 75% on AT&T, 71% on YaleB, and 68% on CelebA, we still can differentiate individuals. Overall, our proposed method shows the advantage in securing the data while retaining high data utility.

## 4.7 Conclusion

In this work, we introduce a mathematical tool  $\epsilon$ -DR to evaluate privacy preserving mechanisms. We also propose a non-linear dimension reduction framework. This framework projects data onto lower dimension domain in which it prevents reconstruction attacks and preserves data utility. The dimension-reduced data can be used effectively for the machine learning tasks such as classification. In our future works, we plan to extend the framework

to adapt with different types of data, such as time series and categorical data. We will apply different metrics to compute the distance other than  $l_2$  norm and investigate the framework on several applications in security systems and data collaborative contributed systems.

## References

- [1] Credit Card Fraud Detection — kaggle.com. <https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud>. [Accessed 16-Dec-2022].
- [2] fetch datasets 2014; Version 0.10.0 — imbalanced-learn.org. [https://imbalanced-learn.org/stable/references/generated/imblearn.datasets.fetch\\_datasets.html](https://imbalanced-learn.org/stable/references/generated/imblearn.datasets.fetch_datasets.html). [Accessed 16-Dec-2022].
- [3] Keel: Software tool. evolutionary algorithms for data mining.
- [4] Maximal and minimal points of functions theory.
- [5] Sphere, Aug 2021.
- [6] Martín Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep Learning with Differential Privacy. (Ccs), 2016.
- [7] Umang Aggarwal, Adrian Popescu, and Celine Hudelot. Active Learning for Imbalanced Datasets. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1417–1426, Snowmass Village, CO, USA, March 2020. IEEE.
- [8] Mohammad Al-rubaie, Pei-yuan Wu, J Morris Chang, and Sun-yuan Kung. Privacy-Preserving PCA on Horizontally-Partitioned Data.
- [9] Pierre Baldi. Autoencoders, Unsupervised Learning, and Deep Architectures. *ICML Unsupervised Transf. Learn.*, pages 37–50, 2012.
- [10] Raphael Bost, Raluca Ada Popa, Stephen Tu, and Shafi Goldwasser. Machine learning classification over encrypted data. 01 2015.

- [11] Raphaël Bost, Ra Popa, Stephen Tu, and S Goldwasser. Machine Learning Classification over Encrypted Data. *Ndss '15*, (February):1–31, 2015.
- [12] Kamalika Chaudhuri. Privacy-preserving logistic regression. pages 1–8.
- [13] Kamalika Chaudhuri and Claire Monteleoni. Privacy-preserving logistic regression. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 289–296. Curran Associates, Inc., 2009.
- [14] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, 16:321–357, June 2002.
- [15] Nitesh V. Chawla, Aleksandar Lazarevic, Lawrence O. Hall, and Kevin W. Bowyer. Smoteboost: Improving prediction of the minority class in boosting. In Nada Lavrač, Dragan Gamberger, Ljupčo Todorovski, and Hendrik Blockeel, editors, *Knowledge Discovery in Databases: PKDD 2003*, pages 107–119, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.
- [16] Hong-You Chen and Wei-Lun Chao. On bridging generic and personalized federated learning for image classification. In *International Conference on Learning Representations*, 2022.
- [17] Xiao Chen, Peter Kairouz, and Ram Rajagopal. Understanding compressive adversarial privacy. *CoRR*, abs/1809.08911, 2018.
- [18] Xiao Chen Lalitha Sankar Ram Rajagopal Chong Huang, Peter Kairouz. Generative Adversarial Privacy. *Privacy in Machine Learning and Artificial Intelligence Workshop, ICML 2018*, 2018.

- [19] Muhammad Enamul Hoque Chowdhury, Tawsifur Rahman, Amith Khandakar, Rashid Mazhar, Muhammad Abdul Kadir, Zaid Bin Mahbub, Khandakar Reajul Islam, Muhammad Salman Khan, Atif Iqbal, Nasser Al-Emadi, and Mamun Bin Ibne Reaz. Can AI help in screening viral and COVID-19 pneumonia? *CoRR*, abs/2003.13145, 2020.
- [20] Wikipedia contributors. Wilcoxon signed-rank test — Wikipedia the free encyclopedia, 2022. Online; accessed 22-December-2022.
- [21] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-Balanced Loss Based on Effective Number of Samples. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9260–9269, Long Beach, CA, USA, June 2019. IEEE.
- [22] Wangzhi Dai, Kenney Ng, Kristen Severson, Wei Huang, Fred Anderson, and Collin Stultz. Generative Oversampling with a Contrastive Variational Autoencoder. In *2019 IEEE International Conference on Data Mining (ICDM)*, pages 101–109, Beijing, China, November 2019. IEEE.
- [23] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.
- [24] Jian-Hui Duan, Wenzhong Li, and Sanglu Lu. *FedDNA: Federated Learning with Decoupled Normalization-Layer Aggregation for Non-IID Data*, pages 722–737. 09 2021.
- [25] Cynthia Dwork. Differential privacy. *Proc. 33rd Int. Colloq. Autom. Lang. Program.*, pages 1–12, 2006.
- [26] F. Emekci, O. D. Sahin, D. Agrawal, and A. El Abbadi. Privacy preserving decision tree learning over multiple parties. *Data Knowl. Eng.*, 63(2):348–361, 2007.



- [27] Seyda Ertekin, Jian Huang, Leon Bottou, and Lee Giles. Learning on the border: active learning in imbalanced data classification. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management - CIKM '07*, page 127, Lisbon, Portugal, 2007. ACM Press.
- [28] Seyda Ertekin, Jian Huang, and C. Lee Giles. Active learning for class imbalance problem. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '07*, page 823, Amsterdam, The Netherlands, 2007. ACM Press.
- [29] Ines Feki, Sourour Ammar, Yousri Kessentini, and Khan Muhammad. Federated learning for COVID-19 screening from Chest X-ray images. *Applied Soft Computing*, 106:107330, July 2021.
- [30] Alberto Fernández, Julián Luengo, Joaquin Derrac, Jesús Alcalá-Fdez, and Francisco Herrera. Implementation and integration of algorithms into the keel data-mining software tool. *Intelligent Data Engineering and Automated Learning - IDEAL 2009*, page 562–569, 2009.
- [31] Karl Pearson F.R.S. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.
- [32] Yonatan Geifman and Ran El-Yaniv. Deep active learning over the long tail. *CoRR*, abs/1711.00941, 2017.
- [33] A.S. Georghiades, P.N. Belhumeur, and D.J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Anal. Mach. Intelligence*, 23(6):643–660, 2001.

- [34] Mathieu Germain, Karol Gregor, Iain Murray, and Hugo Larochelle. Made: Masked autoencoder for distribution estimation. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 881–889, Lille, France, 07–09 Jul 2015. PMLR.
- [35] Daniel Gissin and Shai Shalev-Shwartz. Discriminative active learning, 2019.
- [36] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks. pages 1–9, 2014.
- [37] Dr Saptarsi Goswami. Class Imbalance, SMOTE, Borderline SMOTE, ADASYN, November 2020.
- [38] Haibo He and E.A. Garcia. Learning from Imbalanced Data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, September 2009.
- [39] Hui Han, Wen-Yuan Wang, editor=Huang De-Shuang Mao, Bing-Huan, Xiao-Ping Zhang, and Guang-Bin Huang. "borderline-smote: A new over-sampling method in imbalanced data sets learning". In *Advances in Intelligent Computing*, pages "878–887", "Berlin, Heidelberg", "2005". "Springer Berlin Heidelberg".
- [40] Andrew Hard, Kanishka Rao, Rajiv Mathews, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. Federated learning for mobile keyboard prediction. *CoRR*, abs/1811.03604, 2018.
- [41] Haibo He, Yang Bai, Edwardo A. Garcia, and Shutao Li. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pages 1322–1328, 2008.

- [42] Ehsan Hesamifard, Hassan Takabi, and Mehdi Ghasemi. CryptoDL : Deep Neural Networks over Encrypted Data. pages 1–21, 2017.
- [43] Ehsan Hesamifard, Hassan Takabi, and Mehdi Ghasemi. Cryptodl: Deep neural networks over encrypted data. *CoRR*, abs/1711.05189, 2017.
- [44] Briland Hitaj, Giuseppe Ateniese, and Fernando Perez-Cruz. Deep Models Under the GAN: Information Leakage from Collaborative Deep Learning. 2017.
- [45] Chen Huang, Yining Li, Chen Change Loy, and Xiaoou Tang. Learning Deep Representation for Imbalanced Classification. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5375–5384, Las Vegas, NV, USA, June 2016. IEEE.
- [46] S. Itahara, T. Nishio, Y. Koda, M. Morikura, and K. Yamamoto. Distillation-based semi-supervised federated learning for communication-efficient collaborative training with non-iid private data. *IEEE Transactions on Mobile Computing*, (01):1–1, mar 2021.
- [47] Stefan Jaeger, Sema Candemir, Sameer Antani, Yi-Xiang Wang, Pu-Xuan Lu, and George Thoma. Two public chest x-ray datasets for computer-aided screening of pulmonary diseases. *Quantitative imaging in medicine and surgery*, 4:475–7, 12 2014.
- [48] Paul A. Jensen. Algorithms for constrained optimization. [https://www.me.utexas.edu/~jensen/ORMM/supplements/units/nlp\\_methods/const\\_opt.pdf](https://www.me.utexas.edu/~jensen/ORMM/supplements/units/nlp_methods/const_opt.pdf).
- [49] Chi Jin, Lydia T. Liu, Rong Ge, and Michael I Jordan. On the local minima of the empirical risk. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

- [50] Justin M. Johnson and Taghi M. Khoshgoftaar. Survey on deep learning with class imbalance. *Journal of Big Data*, 6(1):27, December 2019.
- [51] Daniel S. Kermany, Kang Zhang, and Michael H. Goldbaum. Labeled optical coherence tomography (oct) and chest x-ray images for classification. 2018.
- [52] S. Y. Kung. Compressive privacy: From information estimation theory to machine learning [lecture notes]. *IEEE Signal Processing Magazine*, 34(1):94–112, Jan 2017.
- [53] S.Y. Kung. A compressive privacy approach to generalized information bottleneck and privacy funnel problems. *Journal of the Franklin Institute*, 355, 07 2017.
- [54] 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.
- [55] S. Leroueil. Compressibility of Clays: Fundamental and Practical Aspects. *Journal of Geotechnical Engineering*, 122(7):534–543, July 1996.
- [56] Lusi Li, Haibo He, and Jie Li. Entropy-based Sampling Approaches for Multi-Class Imbalanced Problems. *IEEE Transactions on Knowledge and Data Engineering*, 32(11):2159–2170, November 2020.
- [57] Qinbin Li, Yiqun Diao, Quan Chen, and Bingsheng He. Federated learning on non-iid data silos: An experimental study. *CoRR*, abs/2102.02079, 2021.
- [58] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In *MLSys*, 2020.
- [59] Xiaoxiao Li, Meirui JIANG, Xiaofei Zhang, Michael Kamp, and Qi Dou. FedBN: Federated learning on non-IID features via local batch normalization. In *International Conference on Learning Representations*, 2021.

- [60] Pin Lim, Chi Keong Goh, and Kay Chen Tan. Evolutionary cluster-based synthetic oversampling ensemble (eco-ensemble) for imbalance learning. *IEEE Transactions on Cybernetics*, 47(9):2850–2861, 2017.
- [61] Alexander Liu, Joydeep Ghosh, and Cheryl E. Martin. Generative oversampling for mining imbalanced datasets. In Robert Stahlbock, Sven F. Crone, and Stefan Lessmann, editors, *Proceedings of the 2007 International Conference on Data Mining, DMIN 2007, June 25-28, 2007, Las Vegas, Nevada, USA*, pages 66–72. CSREA Press, 2007.
- [62] Lumin Liu, Jun Zhang, Shenghui Song, and Khaled Ben Letaief. Edge-assisted hierarchical federated learning with non-iid data. *CoRR*, abs/1905.06641, 2019.
- [63] Shigang Liu, Yu Wang, Jun Zhang, Chao Chen, and Yang Xiang. Addressing the class imbalance problem in twitter spam detection using ensemble learning. *Computers and Security*, 69:35–49, 2017. Security Data Science and Cyber Threat Management.
- [64] 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.
- [65] Yang Liu, Xiang Li, Xianbang Chen, Xi Wang, and Huaqiang Li. High-Performance Machine Learning for Large-Scale Data Classification considering Class Imbalance. *Scientific Programming*, 2020:1–16, May 2020.
- [66] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.
- [67] Dhruv Mahajan, Ross B. Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. *CoRR*, abs/1805.00932, 2018.

- [68] Hamed Masnadi-Shirazi, Nuno Vasconcelos, and Arya Iranmehr. Cost-sensitive support vector machines. *CoRR*, abs/1212.0975, 2012.
- [69] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In Aarti Singh and Jerry Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR, 20–22 Apr 2017.
- [70] Aditya Menon and Cheng Soon Ong. Linking losses for density ratio and class-probability estimation. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 304–313, New York, New York, USA, 20–22 Jun 2016. PMLR.
- [71] Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. Agnostic federated learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 4615–4625. PMLR, 09–15 Jun 2019.
- [72] Sankha Subhra Mullick, Shounak Datta, and Swagatam Das. Generative Adversarial Minority Oversampling. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1695–1704, Seoul, Korea (South), October 2019. IEEE.
- [73] Tb Pedersen, Y Saygın, and E Savaş. Secret charing vs. encryption-based techniques for privacy preserving data mining. *Fac. Eng. Nat. Sci. Sabanci Univ. Istanbul, TURKEY*, 1:1–11, 2007.
- [74] Nhathai Phan, Yue Wang, Xintao Wu, and Dejing Dou. Differential Privacy Preservation for Deep Auto-Encoders: An Application of Human Behavior Prediction. *Aaai*, pages 1309–1316, 2016.

- [75] Zhicong Qiu, David J. Miller, and George Kesidis. A maximum entropy framework for semisupervised and active learning with unknown and label-scarce classes. *IEEE Transactions on Neural Networks and Learning Systems*, 28(4):917–933, 2017.
- [76] Vivek Kumar Rangarajan Sridhar. Unsupervised Text Normalization Using Distributed Representations of Words and Phrases. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 8–16, Denver, Colorado, 2015. Association for Computational Linguistics.
- [77] M.H. Rashid, J.H. Wang, and C. Li. Convergence analysis of a method for variational inclusions. *Applicable Analysis*, 91(10):1943–1956, October 2012.
- [78] Anit Kumar Sahu, Tian Li, Maziar Sanjabi, Manzil Zaheer, Ameet Talwalkar, and Virginia Smith. On the convergence of federated optimization in heterogeneous networks. *CoRR*, abs/1812.06127, 2018.
- [79] Anit Kumar Sahu, Tian Li, Maziar Sanjabi, Manzil Zaheer, Ameet S. Talwalkar, and Virginia Smith. On the convergence of federated optimization in heterogeneous networks. *ArXiv*, abs/1812.06127, 2018.
- [80] F. S. Samaria and A. C. Harter. Parameterisation of a stochastic model for human face identification. In *Proceedings of 1994 IEEE Workshop on Applications of Computer Vision*, pages 138–142, Dec 1994.
- [81] Chris Seiffert, Taghi M. Khoshgoftaar, Jason Van Hulse, and Amri Napolitano. Rusboost: A hybrid approach to alleviating class imbalance. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 40(1):185–197, 2010.
- [82] Burr Settles and Mark Craven. An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing - EMNLP '08*, page 1070, Honolulu, Hawaii, 2008. Association for Computational Linguistics.

- [83] Adi Shamir. How to share a secret. *Commun. ACM*, pages 612–613, 1979.
- [84] C. E. Shannon. A Mathematical Theory of Communication. *Bell System Technical Journal*, 27(3):379–423, July 1948.
- [85] Li Shen, Zhouchen Lin, and Qingming Huang. Learning deep convolutional neural networks for places2 scene recognition. *CoRR*, abs/1512.05830, 2015.
- [86] Tao Shen, J. Zhang, Xinkang Jia, Fengda Zhang, Gang Huang, Pan Zhou, Fei Wu, and Chao Wu. Federated mutual learning. *ArXiv*, abs/2006.16765, 2020.
- [87] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.
- [88] Yue Tan, Guodong Long, LU LIU, Tianyi Zhou, Qinghua Lu, Jing Jiang, and Chengqi Zhang. Fedproto: Federated prototype learning across heterogeneous clients. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(8):8432–8440, 2022.
- [89] Hao Wang, Zakhary Kaplan, Di Niu, and Baochun Li. Optimizing federated learning on non-iid data with reinforcement learning. In *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, pages 1698–1707, 2020.
- [90] Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. In *International Conference on Learning Representations*, 2020.
- [91] ”Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H.” Vincent Poor. ”tackling the objective inconsistency problem in heterogeneous federated optimization”. *Advances in Neural Information Processing Systems*, ”2020-December”, ”2020”.




- [92] Wikipedia contributors. Receiver operating characteristic — Wikipedia, the free encyclopedia. [https://en.wikipedia.org/w/index.php?title=Receiver\\_operating\\_characteristic&oldid=1081635328](https://en.wikipedia.org/w/index.php?title=Receiver_operating_characteristic&oldid=1081635328), 2022.
- [93] Zhenyu Wu, Zhangyang Wang, Zhaowen Wang, and Hailin Jin. Towards privacy-preserving visual recognition via adversarial training: A pilot study. *CoRR*, abs/1807.08379, 2018.
- [94] Lucila Ohno-Machado Xiaoqian Jiang, Zhanglong Ji, Shuang Wang, Noman Mohammed, Samuel Cheng. Differential-Private Data Publishing Through Component Analysis. *Trans. data Priv.*, 6(1):19–34, 2013.
- [95] Kun Xie, Xueping Ning, Xin Wang, Jigang Wen, Xiaoxiao Liu, Shiming He, editor="Wang-Guojun Zhang, Daqiang", Albert Zomaya, Gregorio Martinez, and Kenli" Li. "an efficient privacy-preserving compressive data gathering scheme in wsns". In "*Algorithms and Architectures for Parallel Processing*", pages "702–715", "Cham", "2015". "Springer International Publishing".
- [96] Yuxi Xie, Min Qiu, Haibo Zhang, Lizhi Peng, and Zhenxiang Chen. Gaussian distribution based oversampling for imbalanced data classification. *IEEE Transactions on Knowledge and Data Engineering*, 34(2):667–679, 2022.
- [97] Qian Ya-Guan, Ma Jun, Zhang Xi-Min, Pan Jun, Zhou Wu-Jie, Wu Shu-Hui, Yun Ben-Sheng, and Lei Jing-Sheng. EMSGD: An Improved Learning Algorithm of Neural Networks With Imbalanced Data. *IEEE Access*, 8:64086–64098, 2020.
- [98] Satya Prakash Yadav, Bhoopesh Singh Bhati, Dharmendra Prasad Mahato, Sachin Kumar, and SpringerLink (Online service). *Federated Learning for IoT Applications*. Springer International Publishing Imprint Springer., Cham, 2022. OCLC: 1295622946.


- [99] M. Yang, T. Zhu, B. Liu, Y. Xiang, and W. Zhou. Machine learning differential privacy with multifunctional aggregation in a fog computing architecture. *IEEE Access*, 6:17119–17129, 2018.
- [100] Andrew Chi-Chih Yao. How to generate and exchange secrets. *27th Annu. Symp. Found. Comput. Sci. (sfcs 1986)*, (1):162–167, 1986.
- [101] Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, Nghia Hoang, and Yasaman Khazaeni. Bayesian nonparametric federated learning of neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7252–7261. PMLR, 09–15 Jun 2019.
- [102] Shaolei Zhai, Xin Jin, Ling Wei, Hongxuan Luo, and Min Cao. Dynamic Federated Learning for GMEC With Time-Varying Wireless Link. *IEEE Access*, 9:10400–10412, 2021.
- [103] Jun Zhang, Zhenjie Zhang, Xiaokui Xiao, Yin Yang, and Marianne Winslett. Functional Mechanism: Regression Analysis under Differential Privacy. pages 1364–1375, 2012.
- [104] Xinwei Zhang, Mingyi Hong, Sairaj V. Dhople, Wotao Yin, and Yang Liu. Fedpd: A federated learning framework with optimal rates and adaptivity to non-iid data. *CoRR*, abs/2005.11418, 2020.
- [105] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *ArXiv*, abs/1806.00582, 2018.
- [106] Shuheng Zhou, Katrina Ligett, and Larry Wasserman. Differential privacy with compression. *CoRR*, 2009.

- [107] Hangyu Zhu, Jinjin Xu, Shiqing Liu, and Yaochu Jin. Federated learning on non-iid data: A survey. *Neurocomputing*, 465:371–390, 2021.
- [108] Di Zhuang, Sen Wang, and J Morris Chang. Fripal: Face recognition in privacy abstraction layer. In *2017 IEEE Conference on Dependable and Secure Computing*, pages 441–448. IEEE, 2017.

## Appendix A: Copyright Permissions

The permission below is for the reproduction of material in Chapter 4.

Home Help ▾ Live Chat Sign in Create Account



**AutoGAN-based dimension reduction for privacy preservation**

**Author:** Hung Nguyen, Di Zhuang, Pei-Yuan Wu, Morris Chang

**Publication:** Neurocomputing

**Publisher:** Elsevier

**Date:** 7 April 2020

© 2019 Elsevier B.V. All rights reserved.

### Journal Author Rights

Please note that, as the author of this Elsevier article, you retain the right to include it in a thesis or dissertation, provided it is not published commercially. Permission is not required, but please ensure that you reference the journal as the original source. For more information on this and on your other retained rights, please visit: <https://www.elsevier.com/about/our-business/policies/copyright#Author-rights>

**BACK**

**CLOSE WINDOW**

© 2023 Copyright - All Rights Reserved | Copyright Clearance Center, Inc. | Privacy statement | Data Security and Privacy | For California Residents  
| Terms and Conditions Comments? We would like to hear from you. E-mail us at [customer@copyright.com](mailto:customer@copyright.com)