Ph.D. DISSERTATION

Learning from Limited Data: Deep Generative Models and Applications

제한된 데이터로부터의 학습: 딥 생성 모델 및 응용

BY

Uiwon Hwang

AUGUST 2023

DEPARTMENT OF ELECTRICAL AND
COMPUTER ENGINEERING
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

Ph.D. DISSERTATION

Learning from Limited Data: Deep Generative Models and Applications

제한된 데이터로부터의 학습: 딥 생성 모델 및 응용

BY

Uiwon Hwang

AUGUST 2023

DEPARTMENT OF ELECTRICAL AND
COMPUTER ENGINEERING
COLLEGE OF ENGINEERING
SEOUL NATIONAL UNIVERSITY

Learning from Limited Data: Deep Generative Models and Applications

제한된 데이터로부터의 학습: 딥 생성 모델 및 응용

지도교수 윤 성 로
이 논문을 공학박사 학위논문으로 제출함

2023년 8월

서울대학교 대학원

전기·정보 공학부

황 의 원

황의원의 공학박사 학위 논문을 인준함

2023년 8월

위 원 장: 원 중 호 (인)

부위원장: 윤 성 로 (인)

위 원: 이 종 환 (인)

위 원: 문 태 섭 (인)

위 원: 오 민 환 (인)

Abstract

Deep learning has made remarkable progress in automatically extracting valuable insights from data. However, when it comes to real-world applications, the scarcity of training data in the real world often presents challenges such as missing values, class and attribute imbalance, and incomplete labels. These challenges introduce discrepancies between real-world data and the assumptions of deep learning models, necessitating labor-intensive preprocessing steps that heavily rely on domain expertise. To this end, it is crucial to develop robust artificial intelligence systems that can effectively learn from limited data.

One promising approach to address these challenges is the utilization of deep generative models (DGMs). DGMs employ deep learning to estimate the underlying data distribution and have made significant advancements in recent years. Building upon the potential of DGMs, particularly generative adversarial networks (GANs), this dissertation aims to address the issues associated with imperfect datasets by devising novel methods and applications in the context of *learning from limited data*.

Specifically, this dissertation focuses on three key research topics: (1) GANs for real-world classification, (2) GANs for unsupervised conditional generation, and (3) the applications of DGMs in the domain of electronic health records.

In the first research topic, we focus on real-world classification, which aims to develop robust classification models that can effectively cope with missing values, class imbalance, and missing label problems in datasets. Previous studies have addressed each of these problems separately by applying machine learning-based preprocessing methods before training classifiers. However, we define these three problems as an “imputation”, and propose a new GAN-based framework named *HexaGAN* that considers the interconnection between these problems.

In the second research topic, we focus on unsupervised conditional generation

(UCG), which aims to perform conditional generation in a completely unsupervised manner. Despite significant advancements in DGMs, conditional generation still requires a large amount of labeled data, which is often not available in real-world datasets. To address this problem, UCG methods identify salient attributes of a dataset and generate data containing those attributes. However, existing UCG models assume that the attributes are balanced and fail to learn imbalanced attributes. To overcome this limitation, we propose *Stein Latent Optimization for GAN (SLOGAN)*, which can robustly learn datasets with imbalanced attributes.

In the last research topic, we applied DGMs to biomedical data to address issues with imperfect datasets such as missing data, class imbalance, and missing labels. Firstly, we present a DGM to predict the amyloid positivity of cognitively normal individuals from proxy measures including structural MRI scans, demographic variables and cognitive scores instead of invasive measurements. Our approach can not only provide inexpensive, non-invasive and accurate diagnostics for preclinical Alzheimer's disease, but also meet real-world requirements for clinical translation of deep learning models including transferability and interpretability. Secondly, we construct HexaGAN with a hint mechanism to predict the survival and clinical interventions such as intubation and supplemental oxygen for COVID-19 patients. Our method outperforms combinations of existing techniques for limited data problems.

Throughout this dissertation, we aim to bridge the gap between deep learning models and real-world applications by focusing on learning from limited data and leveraging the potential of DGMs to address challenges in real-world scenarios. Therefore, it is expected that this dissertation will provide valuable insights into DGMs and contribute to future research on learning from limited data across various fields.

keywords: Learning from Limited Data, Deep Generative Models,
Deep Learning, Biomedical Data Science

student number: 2017-25277

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Data serve as the intrinsic driving force behind learning and advancement. Similar to the learning processes of humans and animals, which amass new knowledge and improve their behaviors by extracting insights from sensory data, machine learning is designed to perform desired tasks through learning from provided data. In this context, deep learning stands out for its proficiency in automatically extracting nonlinear representations from data [16]. Both machine learning and deep learning employ mathematical modeling based on various assumptions to estimate the distribution of data, with optimization achieved by minimizing empirical risk [160]. Ultimately, it can be said that the core objective of deep learning research is to automate the learning process, thus enabling the extraction of valuable insights from an abundance of real-world data.

However, real-world data frequently misaligns with the assumptions underlying machine learning and deep learning models. This discordance necessitates preprocessing to align the collected data with the model's assumptions before initiating the learning process. Preprocessing procedures often demand a considerable degree of domain expertise and labor, sometimes making them infeasible. To address these challenges, researchers have explored the potential of automating preprocessing steps using machine learning and deep learning techniques.

In line with the perspectives of some researchers, we believe that it is possible to harness the data distribution estimated via deep generative models. Building upon this notion, this dissertation unfolds novel perspectives and methodologies that leverage deep generative models to facilitate prediction and generation via *learning from limited data*. Furthermore, this dissertation aims to demonstrate the successful real-world applications of deep generative models in the domain of biomedical data science. The subsequent subsections will provide a comprehensive overview of the research topics, motivations, and significance of this dissertation.

1.1 Deep Generative Models (DGMs)

Generative models are designed to synthesize new data that closely resembles the training data. In contrast to discriminative models that primarily focus on learning the boundaries that separate different classes, generative models aim to estimate the data distribution, making them versatile for various tasks. The Gaussian Mixture Model (GMM) [133] is a popular example of a generative model, representing data as a combination of multiple Gaussian distributions.

With the integration of deep learning, deep generative models (DGMs) have the capacity to learn complex and high-dimensional data distributions. DGMs can be likelihood-based, such as variational autoencoders [83], or likelihood-free, such as Generative Adversarial Networks (GANs) [51]. Exploiting their capability to learn complicated data patterns, DGMs find applications in various domains, including image synthesis [79], text and speech generation [20, 50], and drug discovery [49]. This dissertation specifically focuses on GANs that utilize adversarial learning to estimate data distributions [51]. They consist of a generator and a discriminator, which compete to generate highly realistic data. In Chapter 2, we will delve deeper into the details of GANs.

1.2 Learning from Limited Data

1.2.1 Real-world classification

Deep learning models have demonstrated exceptional, even super-human, performance in image classification tasks [65, 67]. This groundbreaking success has inspired attempts to apply deep learning to a wider range of intricate tasks, including object detection [132], text classification [174], and disease prediction [69]. However, the real-world implementation of classifiers frequently faces obstacles due to imperfections inherent in real-world datasets, collectively termed as the *imperfect dataset problem.* This dissertation focuses on three specific challenges related to imperfect datasets: missing data, class imbalance, and missing labels, as illustrated in Figure 1.1.

The first challenge under consideration is the missing data problem. This is prevalent in datasets used in recommender systems [87] and electronic health records [109]. The process to address the missing data problem, called *imputation*, involves the replacement of missing information within data [159]. However, poor or inappropriate imputation can potentially mislead deep learning-based techniques, causing them to learn incorrect data distributions.

Another significant challenge is the class imbalance problem, where one or more classes are underrepresented relative to the others. This imbalance results in a model that is biased toward the majority class and performs poorly on the minority class. This issue poses considerable challenges in real-world applications, such as fraud identification, diagnosis of rare diseases, and customer churn detection, where precise prediction for the minority class is essential.

In deep learning, the amount of labeled training data significantly affects the performance of the model. Insufficient labeled data gives rise to the missing label problem. This challenge occurs frequently in real-world applications such as natural language models [158] or healthcare systems [13], where the cost of labeling is prohibitively high.

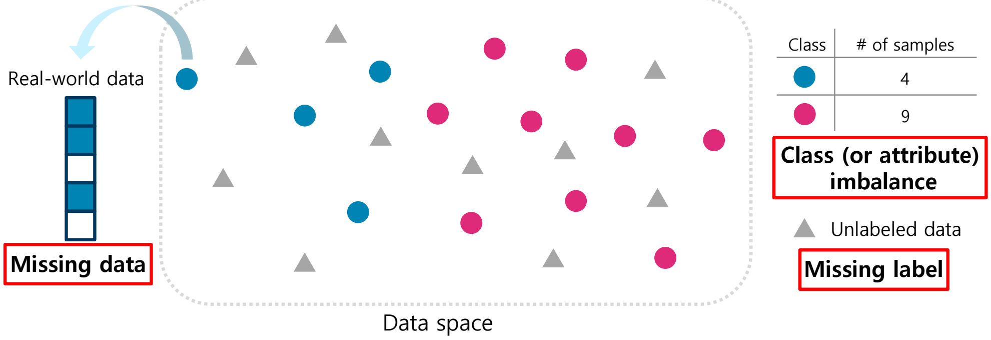


Figure 1.1: Illustrative example of imperfect dataset problem.

In order to address these problems, a preprocessing phase is necessary. However, despite extensive research, no preprocessing technique capable of concurrently addressing all three challenges has not yet been proposed.

1.2.2 Unsupervised conditional generation

Training conditional generative models requires a massive amount of labeled data. However, in many cases, data are often unlabeled or possess only a few labels. To perform conditional generation in an unsupervised manner, several unsupervised conditional GANs have been proposed [27, 116, 120, 9]. In these models, the salient attributes of the data are first identified by unsupervised learning. These models then maximize a lower bound of mutual information between latent codes and generated data, thus clustering the attributes of the underlying data distribution within their latent spaces. These GANs achieve satisfactory performance when the salient attributes of data are balanced.

However, it is important to note that the attributes of real-world data can be *imbalanced*. For example, in the CelebA dataset [101], examples with one attribute (not wearing eyeglasses) outnumber the other attribute (wearing eyeglasses). Similarly, in a biomedical dataset, the number of examples with disease-related attributes might be extremely limited [70]. Thus, the imbalanced nature of real-world attributes must be considered for unsupervised conditional generation. Most of existing unsupervised conditional GANs are not suitable for real-world attributes, because they assume balanced attributes when the imbalance ratio is unknown [27, 116, 120].

1.3 Electronic Health Records (EHRs)

As medical systems become increasingly computerized, electronic health records (EHRs) [57] are contributing greatly to more efficient and systematic medical services compared to previously written medical record systems. One of the important benefits of EHRs is that big data produced from such records can be used for various data science and machine learning studies [106], including statistical analysis of diseases [167], personalized disease prediction [146], and cohort-based disease analysis [122]. However, when we analyze EHRs collected in clinical practice using deep learning models, the imperfect dataset problem can significantly deter the performance of the model. The problems that can occur in data analysis using EHRs are as follows.

Missing data problem Firstly, the dataset under analysis may have missing data. This often happens in longitudinal health records, where the introduction of new attributes such as novel medication or system changes typically begins at specific time points. This leads to bias in missing values towards new attributes [108]. Moreover, medical examination data can have many missing values due to patient-specific requirements or cost constraints [18]. Figure 1.2 illustrates this inherent sparsity in EHR data. Although platforms have recently emerged for automated extraction and organization of data from EHRs [40], the majority of EHR data in clinical practice are still manually entered, leading to potential errors or omissions. Many machine learning algorithms exclude records with missing values from their training dataset.

Class imbalance problem Another challenge that can arise in data analysis using EHRs is the class (or attribute) imbalance problem. This can be observed in real-world

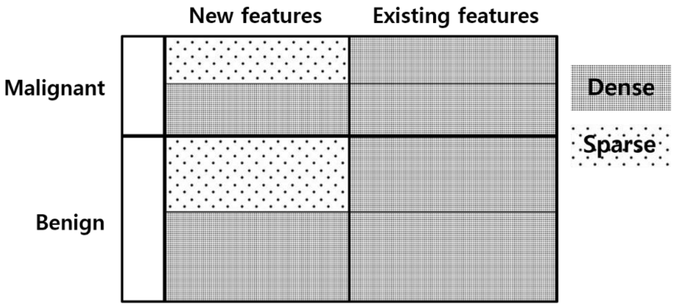


Figure 1.2: Sparseness in electronic health record data. This figure shows a case in which new attributes are added to the health examination items and missing values occur in certain attributes of some records.

scenarios where the number of EHRs for healthy individuals is greater than those for individuals with a specific disease. When models are trained on such highly imbalanced labeled data, they may achieve high accuracy even by classifying all input EHRs as healthy individuals, thus failing to perform meaningful classification. Furthermore, in real-world settings, the attributes within unlabeled EHRs can also be imbalanced. For instance, even among patients with the same disease, the distribution of disease subtypes may be imbalanced. When training deep generative models on such data to synthesize EHRs, the generation performance for the minority subtype can significantly deteriorate.

Missing label problem In EHRs, the absence of class labels, also known as the missing label problem, can occur due to various reasons. Labeling EHRs primarily relies on the judgment of medical professionals, such as disease diagnosis or decisions regarding medical interventions. This labeling process is both time-consuming and expensive. Additionally, labels in EHRs are typically assigned to diseases or interventions that require explicit professional judgment. For this reason, when training deep learning models using data collected from specific equipment, a substantial number of EHRs may have missing labels. Furthermore, diagnostic criteria may vary across different hospitals, or change over time. Consequently, labels assigned based on different diagnostic criteria cannot be directly utilized, and are often treated as missing labels.

We present real-world examples of classification tasks, with a particular focus on EHRs. The following subsections will provide concrete illustrations of the imperfect dataset problem encountered in EHRs.

1.3.1 Preclinical Ahzheimer's disease

Amyloid beta (Aβ) deposition is used as a biomarker in the prevention of Alzheimer's disease since it is a measure of the pathophysiological changes that take place in the preclinical stage of the disease [71, 115]. Aβ deposition can be measured directly using positron emission tomography (PET) or a lumbar puncture; but this is inappropriate for cognitively normal (CN) people because it is costly, time-consuming, and involves exposure to radiation or considerable pain. Proxy measures can be obtained relatively cheaply and safely using structural magnetic resonance imaging (MRI) [5, 14] or tests of the cognitive function [58, 86]. However, the relationships between these measures and the extent of Aβ deposition are complicated.

Deep learning (DL) predicts outcomes from given features by finding the complicated relationships between features and outcomes [94, 107]. In clinical applications, however, the dataset used for training is often imperfect: some features or outcome values are missing, or the number of examples with each outcome is very different. It can be caused by the cost of clinical tests, the cost of physician diagnosis, the prevalence of diseases, or the rarity of hospital visits by healthy people. These problems can hinder the generalization of the model by reducing the amount of training data or introducing biases for specific classes. In addition, different clinicians may diagnose diseases based on different sets of features or labeling criteria, which restricts the use of observations in the new dataset and makes the trained model less efficient or even useless in terms of model utility. Furthermore, the non-linearity of deep neural networks makes it difficult for clinicians to interpret the model's predictions and hence to explain them to patients. Addressing these issues is critical to improving the effectiveness and generalizability of the model.

1.3.2 Coronavirus disease 2019

The coronavirus disease 2019 (COVID-19) has spread around the world since December 2019. The explosion in the number of patients causes a shortage of medical resources including medical staffs and hospital beds [6], so accurate screening of patients who are at high risk of death or who require medical interventions helps efficient allocation of medical resources. In addition, timely clinical interventions, such as intubation and supplemental oxygen, are important to reduce inpatient mortality. However, it is difficult for medical staffs to make decisions about treatment in real time and to allocate medical resources efficiently due to the large number of COVID-19 patients. This has increased the demand for automation of the decision-making process using patients' EHRs. To meet the demand, automated tools, especially deep learning (DL) methods, to predict the risk of death and interventions have been actively proposed [137, 11, 161]. There also exists the imperfect dataset problem in EHRs. For these reasons, DL models for predicting various outcomes related to COVID-19 may not perform best.

1.4 Scope of Dissertation

The rest of this dissertation is organized as follows.

In Chapter 3, we define the missing data, class imbalance, and missing label problems in terms of imputation. Based on insight from imputation, we find out that networks used for imputation can play multiple roles. Moreover, solving the three data problems simultaneously is more effective than solving them in a cascading form. Therefore, we propose a GAN framework consisting of six components to solve the three problems in real world classifications. We derive a new objective function for the imputation of missing data, and demonstrate that it performs better than the existing state-of-the-art imputation methods. We define conditional generation from the perspective of conditional imputation, and confirm that the proposed method works successfully by designing the imputation model to be a part of the framework. In order to deal with the missing label problem, we use semi-supervised learning, in which a classifier generates a synthetic class label for unlabeled data and a discriminator distinguishes fake from real labels.

Chapter 3 is based on the following paper:

- Uiwon Hwang, Dahuin Jung, Sungroh Yoon, "HexaGAN: Generative Adversarial Nets for Real World Classification." in *Proceedings of International Conference on Machine Learning (ICML)*, 2019.

The contributions of Chapter 3 and the proposed HexaGAN are as follows:

1. To the best of our knowledge, this is one of the first studies that defines the three problems (missing data, class imbalance, and missing label) in terms of imputation. Then, we propose HexaGAN to encourage thorough imputation of data with these three problems.

2. To implement real world datasets into existing classifiers, we must apply suitable preprocessing techniques to the datasets. However, our framework is simple to use and works automatically when the absence of data elements and labels is indicated ($\mathbf{m}$ and m_y).

3. We devise a combination of six components and the corresponding cost functions. More specifically, we propose a novel adversarial loss function and gradient penalty for element-wise imputation, confirming that our imputation performance produces stable, state-of-the-art results.

4. In real-world classification, the proposed method significantly outperforms cascading combinations of the existing state-of-the-art methods. As a result, we demonstrate that the components of our framework interplay to solve the problems effectively.

In Chapter 4, we propose unsupervised conditional GANs, referred to as Stein Latent Optimization for GANs (SLOGAN). We define the latent distribution of GANs as Gaussian mixtures to enable the imbalanced attributes to be naturally clustered in a continuous latent space. We derive reparameterizable gradient identities for the mean vectors, full covariance matrices, and mixing coefficients of the latent distribution using Stein's lemma. This enables stable learning and makes latent distribution parameters, including the mixing coefficient, learnable. We then devise a GAN framework with an encoder network and an unsupervised conditional contrastive loss (U2C loss), which can interact well with the learnable Gaussian mixture prior. This framework facilitates the association of data generated from a Gaussian component with a single attribute.

We performed experiments on various real-world datasets. Through experiments, we verified that the proposed method outperforms existing unsupervised conditional GANs in unsupervised conditional generation on datasets with balanced or imbalanced attributes. Furthermore, we confirmed that we could control the attributes to be learned when a small set of probe data is provided.

Chapter 4 is based on the following paper:

- Uiwon Hwang, Heeseung Kim, Dahuin Jung, Hyemi Jang, Hyungyu Lee, Sungroh Yoon, "Stein Latent Optimization for Generative Adversarial Networks." in *Proceedings of International Conference on Learning Representations (ICLR)*, 2022.

The contributions of Chapter 4 and the proposed SLOGAN are as follows:

1. We propose novel Stein Latent Optimization for GANs (SLOGAN). To the best of our knowledge, this is one of the first methods that can perform unsupervised conditional generation by considering the imbalanced attributes of real-world data.

2. To enable this, we derive the implicit reparameterization for Gaussian mixture prior using Stein's lemma. Then, we devise a GAN framework with an encoder

and an unsupervised conditional contrastive loss (U2C loss) suitable for implicit reparameterization.

3. SLOGAN significantly outperforms the existing methods on unsupervised learning tasks, such as cluster assignment, unconditional data generation, and unsupervised conditional generation, on datasets that include balanced or imbalanced attributes.

In Chapter 5, we aim to apply deep generative models to biomedical data which contain the imperfect dataset problem such as missing data, class imbalance, and missing labels.

We modified HexaGAN to predict whether CN individuals are in the preclinical stage (or amyloid positive CN individuals) on the basis of data from structural MRI scans, demographic information, and clinical scores. Our model can make accurate predictions, and its development also embodies three significant steps toward real-world clinical application. Firstly, we trained our model on a dataset with missing features and labels, and with imbalanced classes: some cognitive scores have missing values, amyloid positivities are missing for some participants, and the size of the Aβ+ group is different from that of the A$\beta-$ group. Secondly, we dealt with a situation in which a trained model is required to operate on data collected from different hospitals, in which feature sets or diagnostic criteria may be different, and constructed an appropriate prediction scheme. Thirdly, using explainable artificial intelligence (XAI) techniques, we determined discriminative regions and variables which represent the features that are important in the prediction of early amyloid pathology. Therefore, our considerations for clinical implementation ranging from model architecture to application showed that our model can be successfully used in real-world situations.

We also used a publicly available dataset [30] of chest X-ray images and metadata to predict mortality and interventions of COVID-19 patients. We adopted HexaGAN [70] and additionally applied a hint mechanism [171] to accurately predict the mortality, and whether the patients need intubation or supplemental oxygen. We verified that our method outperforms combinations of existing techniques for limited data problems.

Chapter 5 is based on the following papers:

- Uiwon Hwang*, Sung-Woo Kim*, Dahuin Jung, SeungWook Kim, Hyejoo Lee, Sang Won Seo, JoonKyung Seong, Sungroh Yoon. “Real-world Prediction of Preclinical Alzheimer’s Disease with a Deep Generative Model.” Under review.

- Uiwon Hwang, Euideuk Hwang, Minsoo Kang, Sungroh Yoon, “Prediction of Mortality and Intervention in COVID-19 Patients Using Generative Adversarial Networks.” in *Proceedings of ICML Workshop on Healthcare AI and COVID-19*, 2022.

The contributions of Chapter 5 are summarized as follows:

1. We aim to apply DGMs to EHRs to address imperfect dataset problems such as missing data, class imbalance, and missing label problems.

2. We propose a deep generative model to predict preclinical Alzheimer’s disease in cognitively normal individuals using MRI scans, demographic information and clinical scores. Our model effectively copes with the imperfect dataset problem and significantly improves generalization performances. Our learned model is easily transferable to other hospitals where the feature sets or diagnostic criteria are different from those in the training data. We determine discriminative regions and variables from the population-level attributions we defined.

3. We construct deep generative models to provide an accurate prediction of mortality and interventions for COVID-19 patients from the dataset with limited data problems. To enable this, we use HexaGAN and additionally apply a hint mechanism to enhance the prediction performance. Our method significantly outperforms combinations of existing methods. Especially, our method achieves about twice higher performance (specificity) than benchmark combinations in predicting the risk of death.

In Chapter 6, we will conclude this dissertation by examining potential future research avenues. Building upon the findings and insights gained throughout this dissertation, we will identify areas that require further exploration and propose potential directions for future investigations.

Chapter 2

Background

2.1 Generative Adversarial Networks (GANs)

Generative Adversarial Networks (GANs) implicitly estimate data distribution through a competitive relationship between the generator and the discriminator. The generator creates synthetic data from a latent vector. Simultaneously, the discriminator distinguishes between real and generated data. The error signal from this process enables both the discriminator and generator to learn and refine their abilities, with the goal of producing higher quality synthetic data. An overview of the GAN architecture is illustrated in Figure 2.1.

2.1.1 Adversarial objective

Likelihood-free learning Numerous DGMs employ a maximum likelihood approach. However, high test likelihood does not necessarily guarantee great sample quality. Theis et al. [153] provided two illustrative scenarios. Firstly, we can consider a situation where the likelihood is high but the quality is low. Let us suppose we have a discrete noise mixture model that generates real data with a 1% probability and noise with a 99% probability (i.e., $\log p_\theta(\mathbf{x}) = \log[0.01 p_{\text{data}}(\mathbf{x}) + 0.99 p_{\text{noise}}(\mathbf{x})]$ where p_θ, p_{data}, and p_{noise} denote the distribution of the model, true data, and noise, respectively). Then,

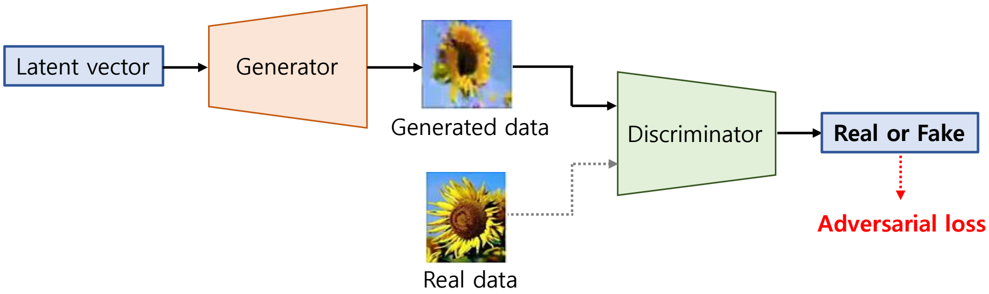


Figure 2.1: Overview of generative adversarial networks.

the expected log-likelihood is bounded as follows:

$$\mathbb{E}_{p_{\text{data}}}[\log p_{\text{data}}(\mathbf{x})] \geq \mathbb{E}_{p_{\text{data}}}[\log p_\theta(\mathbf{x})] \geq \mathbb{E}_{p_{\text{data}}}[\log p_{\text{data}}(\mathbf{x})] - \log 100 \tag{2.1}$$

Despite the low sample quality, the expected log-likelihood increases as the dimensionality increases. This overshadows the relatively small constant value of $\log 100 \simeq 6.64$ bits, resulting in a high test likelihood (i.e., $\mathbb{E}_{p_{\text{data}}}[\log p_\theta(\mathbf{x})] \simeq \mathbb{E}_{p_{\text{data}}}[\log p_{\text{data}}(\mathbf{x})]$). A practical example of Glow [84] is presented in Figure 2.2, illustrating the diminishing effect of the constant with increasing data dimensionality. On the other hand, the second scenario provides an instance where high sample quality does not guarantee high test likelihood. If a model memorizes the training data and generates the exact same data, this model undoubtedly produces high-quality samples because the generated samples are identical to the real data. However, since the model never generates test data, the probability of the test data becomes zero, resulting in a poor test likelihood.

The philosophy behind likelihood-based models postulates that high likelihood would naturally lead to high quality. However, the examples above demonstrate otherwise, indicating that we should pursue these two desiderata separately. Therefore, likelihood-free models, represented by GANs, perform a two-sample test considering the null hypothesis that the samples from the real data distribution and the generated data distribution are identical, using samples from both distributions [114].

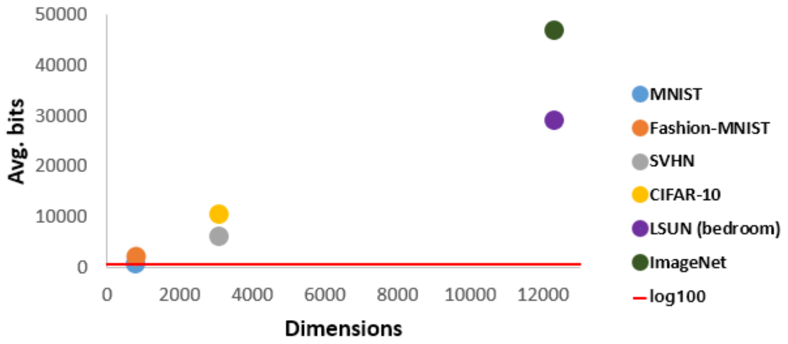


Figure 2.2: Average negative log-likelihood evaluated on various datasets for Glow.

Class probability matching Vanilla GANs [51] test the hypothesis using the density ratio of the true data distribution and the model distribution. To compute the probability of data belonging to the real data, a discriminator $D : \mathbf{x} \rightarrow [0, 1]$, where $\mathbf{x}$ denotes a data point, is introduced. With an assumption that the marginal probabilities of real data and generated data are balanced, the density ratio can be represented as follows:

$$r(\mathbf{x}) = \frac{D(\mathbf{x})}{1 - D(\mathbf{x})} \tag{2.2}$$

Given the discriminator, we can derive the vanilla GAN loss to train the generator G and the discriminator D as follows:

$$\min_G \max_D \mathbb{E}_{\mathbf{x}}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}}[\log(1 - D(G(\mathbf{z}))] \tag{2.3}$$

where $\mathbf{z}$ is a latent vector sampled from a predefined latent distribution. For the optimal discriminator, the training objective for the generator is equivalent to minimizing the Jensen-Shannon divergence (JSD) between the real data distribution and the generated data distribution.

Vanishing gradient on the generator With JSD loss, it is possible to learn an optimal (perfect) discriminator D^* and its gradient will be zero almost everywhere when the support of the real data distribution and the generated data distribution are disjoint or lie on low dimensional manifolds [7]. However, the gradient of the generator approaches zero, resulting in vanishing gradients. Arjovsky and Bottou [7] demonstrated this phenomenon through the following theorem:

Theorem 2.1 (Arjovsky and Bottou [7])**.** *If conditions of Theorems 2.1 or 2.2 in Arjovsky and Bottou [7] are satisfied, $\|D - D^*\| < \epsilon$, and $\mathbb{E}_{\mathbf{z}}[\|J_G G(\mathbf{z})\|_2^2] \leq M^2$, then*

$$\lim_{\|D-D^*\|\to 0} \|\nabla_G \mathbb{E}_{\mathbf{z}}[\log(1 - D(G(\mathbf{z})))]\|_2 < \lim_{\epsilon \to 0} M \frac{\epsilon}{1-\epsilon} = 0 \tag{2.4}$$

Theorem 2.1 verifies that a strong discriminator vanishes gradients on the generator, and hinders the effective learning of the generator.

Beyond Jensen-Shannon divergence To overcome the challenge of vanishing gradients, researchers have explored an alternative approach known as momentum matching. This approach focuses on aligning the moments of the real data distribution and the generated data distribution because distributions with identical moments are considered equivalent. The integral probability metric (IPM) adheres to this principle. IPM finds a critic f, represented as a function within a specific function space $\mathcal{F}$, that maximizes the difference in the mean values of the real and generated data. The computation of this difference can be expressed as follows:

$$\sup_{f \in \mathcal{F}} \mathbb{E}_{p(\mathbf{x})}[f(\mathbf{x})] - \mathbb{E}_{q(\mathbf{z})}[f(G(\mathbf{z}))] \tag{2.5}$$

where $p(\mathbf{x})$ and $q(\mathbf{z})$ represent the real data distribution and the latent distribution, respectively. When the two distributions are dissimilar, the difference in the mean values increases, and when the distributions are similar, the difference decreases. Thus, the critic serves as a measure of the discrepancy between the two distributions.

The Wasserstein distance, also known as the Earth-Mover (EM) distance is a variant of IPM, and utilized by Arjovsky et al. [8] to address the vanishing gradient problem. The Wasserstein distance W between the real data distribution and the generated data distribution can be represented as follows:

$$W(p(\mathbf{x}), q_\theta(\mathbf{x})) = \inf_{\gamma \in \Pi(p^*, q_\theta)} \mathbb{E}_{(\mathbf{x}, \mathbf{x}') \sim \gamma}[||\mathbf{x} - \mathbf{x}'||] \tag{2.6}$$

where $p^* = p(\mathbf{x})$ and $q_\theta = q_\theta(\mathbf{x})$ are the real data distribution and the generated data distribution, respectively. However, the above equation is computationally infeasible because finding an optimal transport plan γ is intractable. To implement the Wasserstein distance, the Kantorovich-Rubinstein duality is employed, which allows us to obtain a similar form to Equation 2.5 as follows:

$$W(p^*, q_\theta) = \sup_{||f||_{\mathrm{Lip}} \leq 1} \mathbb{E}_{\mathbf{x} \sim p^*}[f(\mathbf{x})] - \mathbb{E}_{\mathbf{x} \sim q_\theta}[f(\mathbf{x})] \tag{2.7}$$

where $|| \cdot ||_{\mathrm{Lip}}$ is the Lipschitz constant of a function. The generator, parameterized with θ, is trained to minimize the Wasserstein adversarial loss, while the critic f is trained to maximize the loss. The critic function f plays the role of the discriminator in that it assigns high values to real data samples and low values to generated data samples. In Chapter 3, we will present the expansion of the element-wise adversarial loss for missing data imputation, which is originally derived from the vanilla adversarial loss, to the Wasserstein adversarial loss.

Stable adversarial learning If the critic f is not a 1-Lipschitz function, it fails to accurately measure the discrepancy between the real data distribution and the generated data distribution. This issue is exemplified in Figure 2.3. In the left part of the figure, where f is a 1-Lipschitz function, it exhibits a smooth behavior and indicates a low discrepancy between p^* and q_θ. However, in the right part of the figure, where f is not 1-Lipschitz, it becomes non-smooth and indicates a high discrepancy even when p^* and q_θ are similar. To enforce the Lipschitz constraint, it is crucial to employ an appropriate regularization method for f.

The original Wasserstein GAN paper [8] employed weight clipping as a means

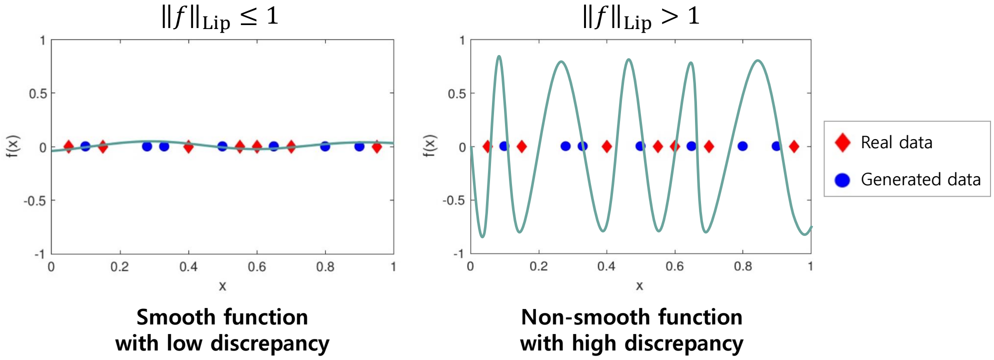


Figure 2.3: Effect of Lipschitz constraint of critic.

of enforcing the Lipschitz constraint, restricting the range of the weights to $W \in [0.01, 0.01]^{d_w}$, where d_w is the dimension of the weights. However, this approach led to weights becoming excessively small, biasing the critic towards overly simplistic functions, and the issues of exploding or vanishing gradients still persisted. To overcome these challenges, various regularization methods for the critic have been proposed [56, 126, 152, 113]. WGAN-GP [56] and WGAN-LP [126] penalize the gradients of the critic to satisfy the Lipschitz constraint and stabilize the training process.

SN-GAN [113] utilizes the upper bound of the spectral norm (also known as the Lipschitz norm) for regularization of the critic. In Wasserstein GANs, the critic typically consists of (leaky) ReLU activations, which satisfy $|a_l|_{\text{Lip}} = 1$, where a_l represents the activation function of the l-th layer among $L+1$ layers. Since the last layer of the critic does not have a nonlinear activation function, the upper bound of the spectral norm for the critic can be computed as follows:

$$\|f\|_L \leq \|\mathbf{h}_L \to W^{L+1}\mathbf{h}_L\|_{\text{Lip}} \cdot \|a_L\|_{\text{Lip}} \cdot \|\mathbf{h}_{L-1} \to W^L\mathbf{h}_{L-1}\|_{\text{Lip}} \tag{2.8}$$

$$\cdots \|a_1\|_{\text{Lip}} \cdot \|\mathbf{h}_0 \to W^1\mathbf{h}_0\|_{\text{Lip}} = \prod_{l=1}^{L+1} \|\mathbf{h}_{l-1} \to W^l\mathbf{h}_{l-1}\|_{\text{Lip}} \tag{2.9}$$

$$= \prod_{l=1}^{L+1} \sigma(W^l) \tag{2.10}$$

where $\mathbf{h}_l$ denotes the activation of the l-th layer, W^l is the weight of the l-th layer, and $\sigma(\cdot)$ is the Lipschitz norm. Miyato et al. [113] normalizes the weight for each layer of the critic with the spectral norm $\sigma(W^l)$, which can be expressed as follows:

$$W^l_{\text{SN}} = W^l / \sigma(W^l) \tag{2.11}$$

Therefore, ensuring that the output of every layer is a 1-Lipschitz function (i.e., $\sigma(W^l_{\text{SN}}) = 1$ in all layers) leads to highly stable training of GANs.

In Chapter 5, we will enhance HexaGAN by incorporating spectral normalization. Since HexaGAN employs element-wise critics, the use of gradient penalty becomes computationally and memory-intensive as the dimensionality of the training data increases. The introduction of spectral normalization has significantly reduced computational and memory requirements, enabling HexaGAN to perform effectively even with high-dimensional biomedical data.

2.1.2 Conditional generation

A conditional GAN is an extension of the traditional GAN framework that incorporates conditioning variables, such as class labels or latent codes, during the training and generative process.

Supervised conditional generation The first proposed supervised conditional GANs [110] introduced the class label condition to the vanilla adversarial objective as follows:

$$\min_G \max_D \mathbb{E}_{\mathbf{x}}[\log D(\mathbf{x}|y)] + \mathbb{E}_{\mathbf{z}}[\log(1 - D(G(\mathbf{z}|y))] \tag{2.12}$$

where y denotes the class label. The implementation involves inputting the class condition y into both the generator and the discriminator. Supervised conditional GANs have evolved by improving the way conditions are incorporated into the model and advancing the loss function. Prominent conditional GAN variants, such as ACGAN [118], projection discriminator [111], and ContraGAN [76], have achieved state-of-the-art performances in conditional image generation tasks. However, these conditional GANs rely on supervised training and necessitate a large amount of labeled data.

Unsupervised conditional generation Unsupervised conditional GANs aim to generate synthetic samples that not only resemble the real data distribution but also align with specific attributes of the data, without explicit supervision. These models can identify the salient attributes present in the data and utilize them as conditioning variables for the generator. During training, unsupervised conditional GANs are commonly trained to maximize the (approximated) mutual information between the latent code and the generated data as follows:

$$\min_G \max_D V_{\mathrm{UCG}}(D, G) = V(D, G) - \lambda I(c; G(\mathbf{z}, c)) \tag{2.13}$$

where $V(D, G)$ represents an adversarial objective, $I(\cdot;\cdot)$ denotes the mutual information, c is a latent code, and λ denotes a hyperparameter. This encourages the generator to produce samples that contain the desired attributes and follow the real data distribution. Figure 2.4 illustrates the common process of unsupervised conditional GANs.

Several models including InfoGAN [27], ClusterGAN [116], Self-conditioned GAN [100], CD-GAN [120], and PGMGAN [9] have been proposed to perform conditional generation in a completely unsupervised manner. However, these models primarily have two drawbacks: (1) Most of these methods embed the attributes in discrete variables, which induces discontinuity among the embedded attributes. (2) Most of them assume uniform distributions of the attributes, and thus fail to learn the imbalance in attributes when the imbalance ratio is not provided. In Chapter 4, we will address the aforementioned limitations by combining GANs with the gradient estimation of the Gaussian mixture prior via Stein's lemma and representation learning on the latent space.

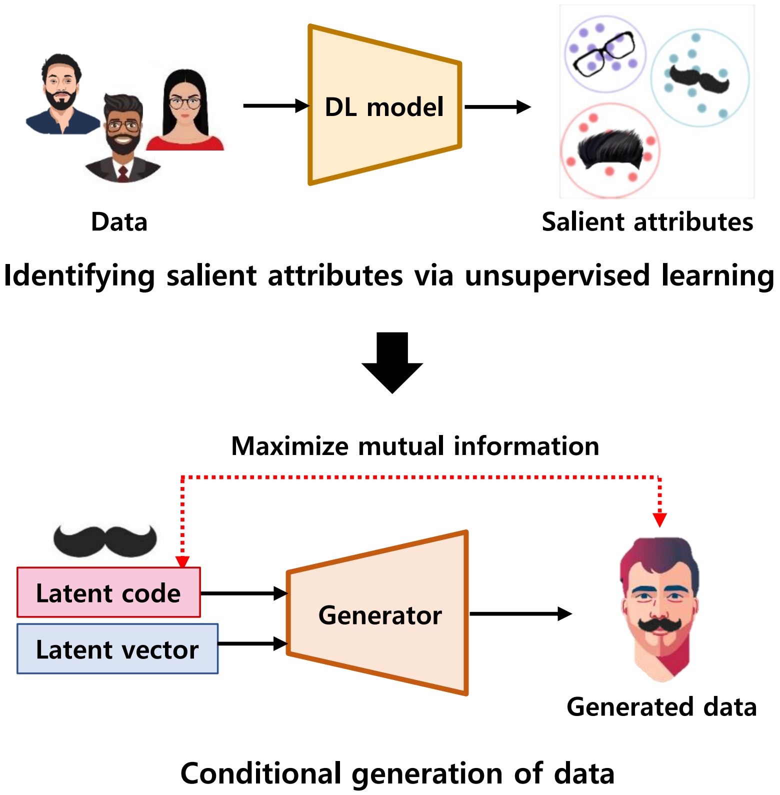

Data
DL model
Salient attributes
Identifying salient attributes via unsupervised learning
Maximize mutual information
Latent code
Latent vector
Generator
Generated data
Conditional generation of data

Figure 2.4: Overview of unsupervised conditional generation.

2.2 Imperfect Dataset Problem

In this dissertation, we define the imperfect dataset problem as comprising several sub-problems including the missing data, class (for labeled data) or attribute (for unlabeled data) imbalance, and missing label problems. To formulate these sub-problems, we consider the set of data $\mathcal{D}$ in the context of binary classification. $\mathcal{D}$ consists of data $\mathbf{x} \in \mathbb{R}^d$, and labels $y \in \{0, 1\}$ where d represents the dimension of the data. We denote the number of data with $y = 1$ and $y = 0$ as n_1 and n_0 respectively. We now introduce two Boolean objects $\mathbf{m} \in \{0, 1\}^d$ and $o \in \{0, 1\}$ that represent the missingness of data and labels, respectively. If an j-th feature or label is missing, m^j or o is set to 0, respectively. With these notations, we can represent the dataset $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{m}_i), (y_i, o_i)\}_{i=1}^N)$, where N is the total number of data in $\mathcal{D}$.

2.2.1 Missing data problem

The missing data problem arises when one or more elements of $\mathbf{m}_i$ are set to 0, indicating the presence of missing values in the dataset. Empirical risk minimization (ERM) relies on the assumption that we can accurately estimate the risk based on training data samples. However, when missing data is present, the empirical risk may no longer provide a reliable estimate of the true risk.

Therefore, it is necessary to impute missing values before using the data. Data-level approaches to handle missing features are to fill them with certain values such as zeros [25] or the average values of each attribute [147]. Filling missing values with zeros assumes no inherent value or information in the missing entries, whereas using the attribute mean provides a more representative estimate based on the available data. Machine learning-based imputation techniques offer more sophisticated approaches to handling missing values. One example is k-nearest neighbors (kNN) imputation [157], where missing values are estimated by considering the values of their nearest neighbors in the data space. Another technique, multivariate imputation by chained equations (MICE) [21] iteratively imputes missing values by regressing each missing feature on the other features and repeating this process multiple times. It captures the relationships between variables and allows for imputations based on conditional distributions. Denoising autoencoders [162, 68] are deep learning models that can be used for imputing missing values. By training an autoencoder on the observed data, it learns to reconstruct the missing entries based on the patterns and structure present in the data.

Taking advantage of the excellent generation capability of GANs, attempts have been made to solve the imperfect dataset problem [171, 139]. GAIN [171] is the first method to use a GAN for imputing missing data. The typical discriminator predicts whether each *instance* is real or fake. However, this task is difficult if all instances have missing data. Instead, GAIN labels each *element* of an instance as missing or not, so that the discriminator can discriminate between real and fake elements. In our experiments, the imputation performance of GAIN with the specific dataset is lower than that of the autoencoder, and the learning curve appears to be unstable.

2.2.2 Class imbalance problem

In an imbalanced dataset, the number of data belonging to one class significantly outnumbers the number of instances belonging to another class (e.g., $n_0 > n_1$). This creates a bias in models toward the majority class. The bias is due to the ERM principle trying to minimize the overall error, which is mostly influenced by the majority class. Consequently, the performance of the minority class tends to be poor as it receives less attention and consideration during training.

To address this issue, various techniques have been developed to mitigate the impact of class imbalance and improve the performance of the minority class. Data-level approaches include resampling methods, which involve either oversampling the minority class (e.g., duplicating instances) or undersampling the majority class (e.g., removing instances) to create a balanced distribution. Machine learning-based oversampling techniques were also proposed, with one notable example being the synthetic minority oversampling technique (SMOTE) [24]. SMOTE generates synthetic samples by interpolating feature vectors between existing minority class instances. In addition, regularization approaches include the use of specialized loss functions or regularization methods. Cost-sensitive learning [144] assigns different misclassification costs to different classes based on their prevalence or importance. Class rectification loss (CRL) [38] is another regularization method that penalizes the model to adjust distances of positive and negative pairs of the minority class. However, these methods have limitations, such as overfitting or increased memory and time requirements for learning, as summarized by Elrahman and Abraham [41].

In contrast, GAN-based methods offer an alternative approach by training a generative model to learn the underlying distribution of the data and generate synthetic samples [43, 103]. By synthesizing new minority class instances, GANs can effectively balance the class distribution and alleviate the class imbalance problem.

2.2.3 Missing label problem

The missing label problem is another violation of the assumption of complete data, represented by o_i being 0. In a supervised learning setting, the standard assumption is that we have complete pairs of $(\mathbf{x}_i, y_i)$ for every instance. Missing labels can lead to difficulties in model training and evaluation, as the model cannot learn directly from incomplete or unlabeled instances. This can result in the model overfitting to the labeled part of the data, where it learns to memorize the available labels without properly generalizing to new, unlabeled instances.

One common approach in semi-supervised learning is the regularization approach, which adds a regularization loss term to the standard supervised learning objective. The regularization term is designed based on the assumption that neighboring or similar instances are likely to share the same label [163, 89, 53, 112, 150]. This encourages the model to produce smooth decision boundaries and make predictions that are consistent with the local data structure. Label propagation methods [163] leverage the notion of graph-based relationships among instances. By propagating labels from labeled instances to their neighboring unlabeled instances, label propagation methods can assign pseudo-labels to the unlabeled data points, effectively incorporating them into the learning process. Entropy minimization techniques [53] aim to reduce the uncertainty in model predictions on the unlabeled instances. By minimizing the entropy of the model's predicted probability distribution, the model is encouraged to produce confident and more reliable predictions, especially in instances that have high predictive uncertainty.

Unlike the regularization approach, the generative approach enhances the performance of a classifier by utilizing raw unlabeled data in training the generative model [85, 2, 136, 141, 33]. This approach capitalizes on the power of GANs to generate synthetic samples and the interaction between the classifier, generator, and discriminator. In particular, TripleGAN [95] is a GAN for semi-supervised learning in which a

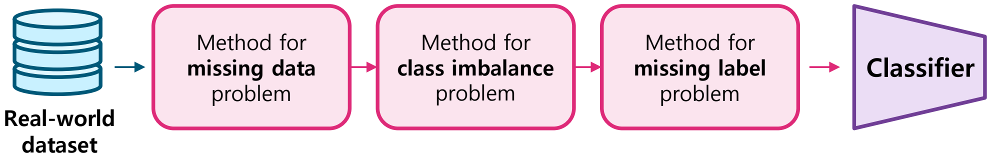

Real-world dataset
Method for missing data problem
Method for class imbalance problem
Method for missing label problem
Classifier

Figure 2.5: Cascade combination of existing machine learning methods to address imperfect dataset problems.

classifier, a generator, and a discriminator interact. The classifier creates pseudo-labels for unlabeled data, and image-label pairs are then passed to the discriminator. The classifier and discriminator are trained competitively.

The approaches mentioned above can only deal with one of the imperfect dataset problems. When confronted with multiple subproblems simultaneously within a training dataset, a cascade combination of various methods can be employed as shown in Figure 2.5. This approach involves constructing a pre-processing pipeline tailored to address the specific problems present in the training data. However, it is important to note that this cascade combination approach often overlooks the potential connections and interdependencies between the different problems. In chapter 3, we will present a GAN framework named *HexaGAN*, which deals with the subproblems simultaneously through interaction between its components.

2.3 Gradient Estimation for Gaussian Mixture

2.3.1 Explicit reparameterization

To effectively train the GAN on a diverse and limited amount of data, the latent distribution of a GAN can be constructed as a Gaussian mixture, and its parameters can be jointly learned with the GAN [59]. One example of this approach is DeLiGAN [59], which incorporates a Gaussian mixture prior and utilizes explicit reparameterization [83] to estimate gradients for the parameters of the Gaussian mixture.

Explicit reparameterization is a straightforward method for estimating gradients of Gaussian mixture parameters. This technique involves introducing an auxiliary noise variable, denoted as ϵ, and performing ancestral sampling to obtain a latent vector. Initially, a component ID, c, is selected based on the mixture weight $p(c)$, which follows a categorical distribution. When the c-th Gaussian component is selected, the latent variable $\mathbf{z}$ is computed as follows:

$$\mathbf{z} = \boldsymbol{\mu}_c + \boldsymbol{\epsilon} \cdot \boldsymbol{\Sigma}_c^{1/2}, \;\; \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \tag{2.14}$$

where $\boldsymbol{\mu}_c$ and $\boldsymbol{\Sigma}_c$ represent the mean vector and the covariance matrix of the c-th component of the Gaussian mixture, respectively.

This reparameterization allows for the backpropagation of gradients into the Gaussian mixture parameters. Importantly, the derivatives of the loss function only update the parameters of the *selected* (c-th) component:

$$\frac{\partial \mathcal{L}(\mathbf{z})}{\partial \boldsymbol{\mu}} = \frac{\partial \mathcal{L}(\mathbf{z})}{\partial \boldsymbol{\mu}_c}, \;\; \frac{\partial \mathcal{L}(\mathbf{z})}{\partial \boldsymbol{\Sigma}} = \frac{\partial \mathcal{L}(\mathbf{z})}{\partial \boldsymbol{\Sigma}_c} \tag{2.15}$$

Although gradient estimation using explicit reparameterization is unbiased, it suffers from high variance. In practice, the Gaussian mixture parameters are learned using a batch average of the estimated gradients, known as stochastic gradient estimation, rather than an expectation over the entire Gaussian mixture. Consequently, explicit reparameterization can lead to unstable and slow convergence of the model.

2.3.2 Implicit reparameterization

The key to successful stochastic gradient estimation for a Gaussian mixture is ensuring an unbiased and low-variance approach. In pursuit of this goal, implicit reparameterization has been studied and extended to various distributions [54, 46]. Implicit reparameterization can be derived from Stein's lemma, which provides a first-order gradient identity for a Gaussian distribution. The univariate case of Stein's lemma can be described as follows:

Lemma 2.1 (Stein [143]). *Let function $h(\cdot) : \mathbb{R} \mapsto \mathbb{R}$ be continuously differentiable. $q(z)$ is a univariate Gaussian distribution parameterized by the mean μ and variance σ. Then, the following identity holds:*

$$\mathbb{E}_{q(z)}\left[\sigma^{-1}(z-\mu)h(z)\right] = \mathbb{E}_{q(z)}\left[\nabla_z h(z)\right] \tag{2.16}$$

Bonnet's theorem [17] and Price's theorem [128], which can be derived from Stein's lemma, enable implicit reparameterization of the Gaussian mean vector and covariance matrix, respectively. Notably, Lin et al. [98] generalized Stein's lemma to exponential family mixtures and linked it to implicit reparameterization. Additionally, Stein's lemma has been applied to various fields of deep learning, including Bayesian deep learning [97] and adversarial robustness [165].

For a single latent vector $\mathbf{z}$, implicit reparameterization [46] updates the parameters of *all* the latent components. Gradient estimation using implicit reparameterization is unbiased and has a lower variance, which enables a more stable and faster convergence of the model.

In Chapter 4, we will present a novel GAN, called *SLOGAN*. To the best of our knowledge, our work is the first to apply Stein's lemma to GANs. The gradients for the parameters of the Gaussian mixture prior in SLOGAN are implicitly reparameterizable. We also expand the use of Gaussian mixture prior to unsupervised conditional generation.

2.4 Representation Learning

Representation learning is a fundamental concept in machine learning that focuses on discovering meaningful semantics in datasets [16]. Perceptually salient and semantically meaningful representations can induce better performance in downstream tasks.

2.4.1 Contrastive learning

Contrastive learning has emerged as a highly effective and widely recognized method within the domain of self-supervised representation learning. It involves learning representations by contrasting neighboring instances with non-neighboring instances [60]. The goal of contrastive learning is to maximize the similarity between positive pairs (neighboring instances) and minimize the similarity between negative pairs (non-neighboring instances). In general, contrastive learning considers data samples $X = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ and their transformed counterparts $X' = \{\mathbf{x}'_1, \dots, \mathbf{x}'_N\}$. A critic function, denoted as $f(\mathbf{x}, \mathbf{x}')$, is defined to approximate the log density ratio $\log p(\mathbf{x}'|\mathbf{x})/p(\mathbf{x}')$:

$$f(\mathbf{x}, \mathbf{x}') \propto \log \frac{p(\mathbf{x}'|\mathbf{x})}{p(\mathbf{x}')} \tag{2.17}$$

This critic function captures the similarity between a data sample and its transformed version.

Zhong et al. [178] derived the contrastive loss from the lower bound of the mutual information between X and X' as follows:

$$I(X; X') = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{N} p(\mathbf{x}'_j|\mathbf{x}_i) \frac{p(\mathbf{x}'|\mathbf{x})}{p(\mathbf{x}')} \tag{2.18}$$

$$\geq \log N + \frac{c_0}{N} \sum_{i=1}^{N} \frac{f(\mathbf{x}_i, \mathbf{x}'_i)}{\sum_{k=1}^{N} f(\mathbf{x}_i, \mathbf{x}'_k)} \tag{2.19}$$

where c_0 is constant. When we define $-\sum_{i=1}^{N} \frac{f(\mathbf{x}_i,\mathbf{x}'_i)}{\sum_{k=1}^{N} f(\mathbf{x}_i,\mathbf{x}'_k)}$ as the contrastive loss, the lower bound of the mutual information $I(X; X')$ is approximately maximized by minimizing the contrastive loss [127]. Several studies have shown that contrastive loss is advantageous for the representation learning of imbalanced data [75, 74, 166].

2.4.2 Representation learning in GANs

Within the context of GANs, representation learning becomes crucial to enable the learned latent representations to possess semantic meaning without relying on labeled training data. GANs map a simple latent distribution into a complex data manifold, and the learned latent space can be highly entangled. However, for downstream tasks, it is desirable to disentangle the latent space and obtain semantically meaningful representations.

Various approaches have been proposed to achieve representation learning in GANs. For example, models such as BiGAN [37], ALI [39], and their variants [72, 15] incorporate an additional encoder network within the GAN framework. This encoder network serves as an inversion mapping of the generator, allowing for the learning of useful feature representations that can benefit supervised tasks. However, without supervision, the learned generator may not perform well in conditional generation tasks.

Representation learning plays a vital role in the proposed methods throughout this dissertation. In Chapters 3 and 5, the encoder network is employed to extract meaningful information from the data. The encoded vector is then fed into the generator to fill in missing values. This approach is especially beneficial for oversampling minority classes by generating only the low-dimensional encoded vectors and feeding them into the generator instead of directly generating high-dimensional data. In Chapter 4, we will present a contrastive loss that cooperates with a learnable latent distribution. The objective of the contrastive loss is to maximizes the mutual information between data and a latent code, which aligns with the concept of unsupervised conditional generation mentioned in Section 2.1.2. To implement the contrastive loss, we adopt an additional encoder to extract information about the latent code, facilitating effective representation learning.

Chapter 3

GANs for Real-world Classification

3.1 Introduction

As deep learning models have achieved super-human performance in image classification tasks [65], there have been increasing attempts to apply deep learning models to more complicated tasks such as object detection [132], text classification [174], and disease prediction [69]. However, real world data are often *dirty*, which means that the elements and labels are missing, or there is an imbalance between different classes of data. This prevents a classifier from being fully effective, and thus a preprocessing phase is required. Despite a considerable amount of research, no preprocessing technique has been proposed to address these three problems concurrently. Therefore, we first propose a framework which deals robustly with dirty data.

The types of data which are typically bedeviled with missing information include the user data employed in recommender systems [87], and in electronic health records [109] utilizing deep learning based classifiers. Rubin [135] identifies three main types of missing data: 1) Data are missing completely at random (MCAR). This type of missing data has no pattern which can be correlated with any other variable, whether observed or not. 2) Data are missing at random (MAR). In this case, the pattern of missing data can be correlated with one or more observed variables. 3) Data are missing (but) not at

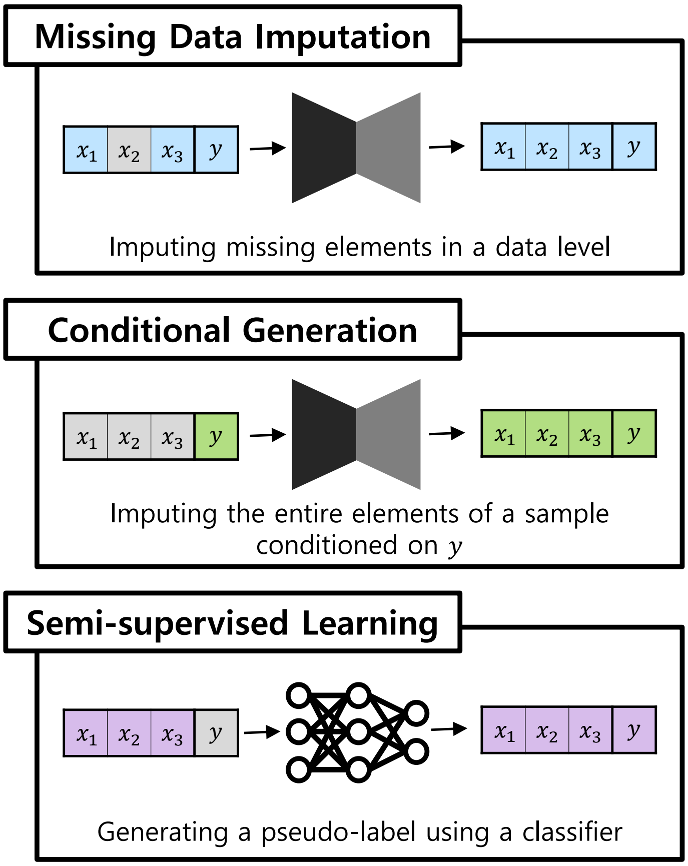


Figure 3.1: Tasks for the three main problems in real world classification. We define missing data imputation as a task that fills in missing data elements. Conditional generation can be defined as a task that imputes the entire elements in an instance conditioned on a certain class. Semi-supervised learning can be defined as a task that imputes missing labels.

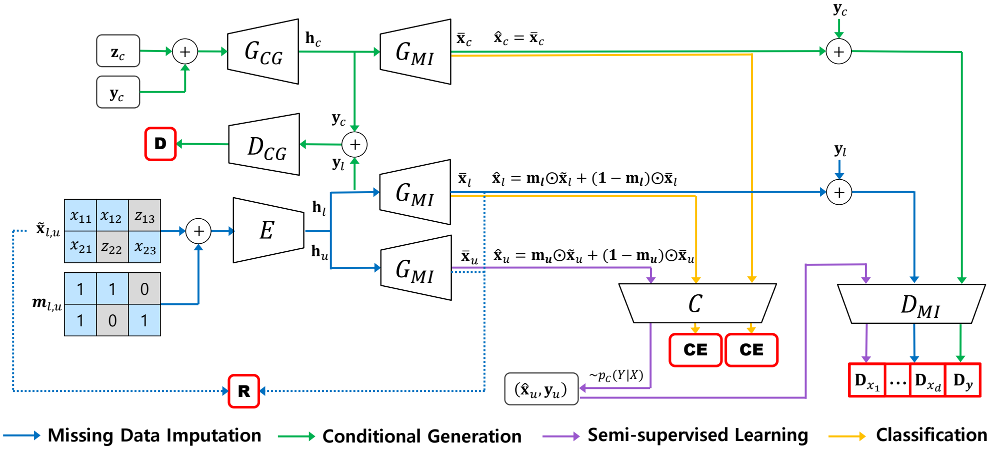

$\mathbf{z}_c$
$\mathbf{y}_c$
G_{CG}
$\mathbf{h}_c$
G_{MI}
$\bar{\mathbf{x}}_c$
$\hat{\mathbf{x}}_c = \bar{\mathbf{x}}_c$
$\mathbf{y}_c$
D
D_{CG}
$\mathbf{y}_c$
$\mathbf{y}_l$
G_{MI}
$\bar{\mathbf{x}}_l$
$\hat{\mathbf{x}}_l = \mathbf{m}_l \odot \tilde{\mathbf{x}}_l + (\mathbf{1} - \mathbf{m}_l) \odot \bar{\mathbf{x}}_l$
$\mathbf{y}_l$
$\tilde{\mathbf{x}}_{l,u}$
x_{11} x_{12} z_{13}
x_{21} z_{22} x_{23}
E
$\mathbf{h}_l$
$\mathbf{h}_u$
G_{MI}
$\bar{\mathbf{x}}_u$
$\hat{\mathbf{x}}_u = \mathbf{m}_u \odot \tilde{\mathbf{x}}_u + (\mathbf{1} - \mathbf{m}_u) \odot \bar{\mathbf{x}}_u$
C
D_{MI}
$\boldsymbol{m}_{l,u}$
1 1 0
1 0 1
CE
CE
$\sim p_C(Y|X)$
$(\hat{\mathbf{x}}_u, \mathbf{y}_u)$
R
$\mathbf{D}_{x_1}$ … $\mathbf{D}_{x_d}$ $\mathbf{D}_y$
Missing Data Imputation
Conditional Generation
Semi-supervised Learning
Classification

Figure 3.2: Overview of the HexaGAN model. Subscripts l, u, and c indicate that a vector is from labeled data, unlabeled data, and class-conditional data respectively. $\tilde{\mathbf{x}}$ denotes a data instance whose missing elements are replaced with noise. $\bar{\mathbf{x}}$ denotes a data generated by G_{MI}. $\hat{\mathbf{x}}$ denotes a data instance whose missing elements are filled with the generated values. $\mathbf{y}$ is a class label. Unlike $\mathbf{y}_l$ and $\mathbf{y}_c$, $\mathbf{y}_u$ is produced by C. $\mathbf{m}$ is a vector that indicates whether corresponding elements are missing or not. $\mathbf{h}$ is a vector in the hidden space. **R** is the reconstruction loss. **D** is the adversarial loss function between G_{CG} and D_{CG}. $\mathbf{D}_{x_i}$ represents the element-wise adversarial loss function. $\mathbf{D}_y$ represents the adversarial loss function for the label. **CE** represents the cross-entropy loss.

random (MNAR). The pattern of this type of missing data can be related to both observed and unobserved variables. We are concerned with MCAR data. The replacement of missing information within data is called *imputation* [159]. Imputation techniques include matrix completion [63], k-nearest neighbors [157], multivariate imputation by chained equations (MICE) [21], denoising autoencoders [162], and methods based on generative adversarial networks (GAN) [171, 139]. Poor or inappropriate imputation can mislead deep learning based techniques into learning the wrong data distribution.

Many real world datasets such as those related to anomaly detection [23] and disease prediction [81] involve poorly balanced classes. The class imbalance problem can be overcome by techniques such as the synthetic minority oversampling technique (SMOTE) [24] and adaptive synthetic (ADASYN) sampling [64]. However, oversampling from the entire data distribution requires a large amount of memory. Cost sensitive loss [144] is also used to solve the class imbalance problem by differentiating cost weights to each class. However, cost sensitive loss tends to overfit to the minority classes [42]. We overcome the class imbalance problem by training a deep generative model to follow the true data distribution, and then generate samples of minority classes for each batch. This requires conditional generation, which we regard as imputation, as shown in Figure 3.1, in which entire elements are imputed according to the appropriate class label.

In deep learning, the amount of labeled training data has a significant impact on the performance. Insufficiency of labeled data is referred to as the missing label problem. It is encountered in real world applications such as natural language models [158] or healthcare systems [13], where the cost of labeling is expensive. Related researchers have proposed semi-supervised methods by which to leverage unlabeled data. Semi-supervised learning is designed to make the best use of unlabeled data, using regularization and generative approaches. The regularization approach adds a regularization loss term, which is designed on the assumption that adjacent data points or the same architectural data points are likely to have the same label [163, 89, 53, 112, 150]. Unlike the regularization approach, the generative approach enhances the performance of a classifier by utilizing raw unlabeled data in training the generative model [85, 2, 136, 141, 33].

As depicted in Figure 3.1, we define the missing data, class imbalance, and missing label problems in terms of imputation. Based on insight concerning the imputation, we find out that networks used for imputation can play multiple roles. Moreover, solving the three data problems simultaneously is more effective than solving them in a cascading form. We propose a GAN framework consisting of six components to solve the three problems in real world classifications. We derive a new objective function for the imputation of missing data, and demonstrate that it performs better than the existing state-of-the-art imputation methods. We define conditional generation from the perspective of conditional imputation, and confirm that the proposed method works successfully by designing the imputation model to be a part of the framework. In order to deal with the missing label problem, we use semi-supervised learning, in which a classifier generates a synthetic class label for unlabeled data and a discriminator distinguishes fake from real labels.

In summary, our contributions are as follows:

- To the best of our knowledge, this is one of the first studies that defines the three problems (missing data, class imbalance, and missing label) in terms of imputation. Then, we propose HexaGAN to encourage thorough imputation of data with these three problems.

- To implement real world datasets into existing classifiers, we must apply suitable preprocessing techniques to the datasets. However, our framework is simple to use and works automatically when the absence of data elements and labels is indicated ($\mathbf{m}$ and m_y).

- We devise a combination of six components and the corresponding cost functions. More specifically, we propose a novel adversarial loss function and gradient penalty for element-wise imputation, confirming that our imputation performance produces stable, state-of-the-art results.

- In real world classification, the proposed method significantly outperforms cascading combinations of the existing state-of-the-art methods. As a result, we demonstrate that the components of our framework interplay to solve the problems effectively.

3.2 HexaGAN

The HexaGAN framework is comprised of six components, as illustrated in Figure 3.2:

- E: the encoder, that transfers both labeled and unlabeled instances into the hidden space.

- G_{MI}: a generator that imputes missing data.

- D_{MI}: a discriminator for missing imputation, that distinguishes between missing and non-missing elements and labels.

- G_{CG}: a generator that creates conditional hidden vectors $\mathbf{h}_c$.

- D_{CG}: a discriminator for conditional generation, that determines whether a hidden vector is from the dataset or has been created by G_{CG}.

- C: the classifier, that estimates class labels. This also works as the label generator.

HexaGAN operates on datasets containing instances $\mathbf{x}^1, ..., \mathbf{x}^n \in \mathbb{R}^d$, where n is the number of instances and d is the number of elements in an instance. The i-th element in a single instance x_i^j is a scalar, and some of these elements may be missing. The first n_l instances are labeled data, and the remaining $n - n_l$ instances are unlabeled data. There are class labels $\mathbf{y}^1, ..., \mathbf{y}^{n_l} \in \mathbb{R}^{n_c}$ corresponding to each instance, where n_c is the number of classes. Boolean vectors $\mathbf{m}^1, ..., \mathbf{m}^n \in \mathbb{R}^d$ indicate whether each element in an instance is missing or not. If m_i^j (the i-th element of a vector $\mathbf{m}^j$) is 0, x_i^j is missing. The boolean $m_y \in \mathbb{R}$ indicates whether an instance has a label or not. If m_y is 0, the label is missing. Thus, labeled instances exist as a set of $D_l = \{(\mathbf{x}^j, \mathbf{y}^j, \mathbf{m}^j, m_y^j = 1)\}_{j=1}^{n_l}$, and unlabeled instances exist as a set of $D_u = \{(\mathbf{x}^j, \mathbf{m}^j, m_y^j = 0)\}_{j=n_l+1}^{n}$.

3.2.1 Missing data imputation

Missing data imputation aims to fill in missing elements using the distribution of data represented by the generative model. In HexaGAN, missing data imputation is performed by E, G_{MI}, and D_{MI}. An instance received by D_{MI} is not labeled as real or fake, but each *element* is labeled as real (non-missing) or fake (missing).

From now on, we omit the superscript for a clearer explanation (i.e., x_i^j=x_i). First, we make a noise vector $\mathbf{z} \in \mathbb{R}^d$ with the same dimension as an input instance $\mathbf{x} \in$

$(\mathbf{x}_l \cup \mathbf{x}_u)$ by sampling from a uniform distribution $U(0, 1)$. We replace the missing elements in the instance with elements of $\mathbf{z}$ to generate $\tilde{\mathbf{x}}$:

$$\tilde{\mathbf{x}} = \mathbf{m} \odot \mathbf{x} + (\mathbf{1} - \mathbf{m}) \odot \mathbf{z} \tag{3.1}$$

where $\odot$ is element-wise multiplication. The objective of our framework is to sample the patterns stored in the model that are the most suitable replacements for the missing data (i.e., to generate samples which follow $p(\mathbf{x}|\tilde{\mathbf{x}}, \mathbf{m})$). Then, $\tilde{\mathbf{x}}$ is concatenated with $\mathbf{m}$, and from the pair $(\tilde{\mathbf{x}}, \mathbf{m})$, the encoder E generates a hidden variable $\mathbf{h} = E(\tilde{\mathbf{x}}, \mathbf{m})$ in the hidden space, which has the dimension d_{h}.

The G_{MI} receives $\mathbf{h}$ and generates $\bar{\mathbf{x}} = G_{MI}(\mathbf{h})$. The missing elements in the input instance are imputed with the generated values, resulting in $\hat{\mathbf{x}}$ as follows:

$$\hat{\mathbf{x}} = \mathbf{m} \odot \mathbf{x} + (\mathbf{1} - \mathbf{m}) \odot \bar{\mathbf{x}} \tag{3.2}$$

The D_{MI} now determines whether each element of the pair $(\hat{\mathbf{x}}, \mathbf{y})$ is real or fake. The label for $\hat{\mathbf{x}}$ is $\mathbf{m}$. The D_{MI} calculates the adversarial losses by determining whether the missingness is correctly predicted for each element, which is then used to train E, G_{MI}, and D_{MI}. The adversarial loss $\mathcal{L}_{G_{MI}}$ which is used to train E and G_{MI}, and $\mathcal{L}_{D_{MI}}$ which is used to train D_{MI} can be expressed as follows:

$$\mathcal{L}_{G_{MI}} = -\sum_{i=1}^{d} \mathbb{E}_{\hat{\mathbf{x}},\mathbf{y},\mathbf{m}} \left[(1 - m_i) \cdot D_{MI}(\hat{\mathbf{x}}, \mathbf{y})_i\right] \tag{3.3}$$

$$\begin{aligned} \mathcal{L}_{D_{MI}} = \sum_{i=1}^{d} & \mathbb{E}_{\hat{\mathbf{x}},\mathbf{y},\mathbf{m}} \left[(1 - m_i) \cdot D_{MI}(\hat{\mathbf{x}}, \mathbf{y})_i\right] \\ & - \mathbb{E}_{\hat{\mathbf{x}},\mathbf{y},\mathbf{m}} \left[m_i \cdot D_{MI}(\hat{\mathbf{x}}, \mathbf{y})_i\right] \end{aligned} \tag{3.4}$$

where $D_{MI}(\cdot)_i$ is the i-th output element of D_{MI}. The following theorem confirms that the proposed adversarial loss functions make the generator distribution converge to the desired data distribution.

Algorithm 3.1 Missing data imputation

```
input  : $\mathbf{x}$ - data with missing values sampled from $D_l$ and $D_u$;
         $\mathbf{m}$ - vector indicating whether elements are missing;
         $\mathbf{z}$ - noise vector sampled from $U(0, 1)$
output : $\hat{\mathbf{x}}$ - imputed data
  repeat
    Sample a batch of pairs $(\mathbf{x}, \mathbf{m}, \mathbf{z})$
    $\tilde{\mathbf{x}} \leftarrow \mathbf{m} \odot \mathbf{x} + (\mathbf{1} - \mathbf{m}) \odot \mathbf{z}$
    $\mathbf{h} \leftarrow E(\tilde{\mathbf{x}}, \mathbf{m})$
    $\bar{\mathbf{x}} \leftarrow G_{MI}(\mathbf{h})$
    $\hat{\mathbf{x}} \leftarrow \mathbf{m} \odot \mathbf{x} + (\mathbf{1} - \mathbf{m}) \odot \bar{\mathbf{x}}$
    Update $D_{MI}$ using stochastic gradient descent (SGD)
    $\nabla_{D_{MI}} \mathcal{L}_{D_{MI}} + \lambda_1 \mathcal{L}_{\mathrm{GP}_{MI}}$
    Update $E$ and $G_{MI}$ using SGD
    $\nabla_E \mathcal{L}_{G_{MI}} + \alpha_1 \mathcal{L}_{\mathrm{recon}}$
    $\nabla_{G_{MI}} \mathcal{L}_{G_{MI}} + \alpha_1 \mathcal{L}_{\mathrm{recon}}$
  until training loss is converged
```

Theorem 3.1. *A generator distribution $p(\mathbf{x}|\mathbf{m}_i = 0)$ is a global optimum for the min-max game of G_{MI} and D_{MI}, if and only if $p(\mathbf{x}|\mathbf{m}_i = 1) = p(\mathbf{x}|\mathbf{m}_i = 0)$ for all $\mathbf{x} \in \mathbb{R}^d$, except possibly on a set of zero Lebesgue measure.*

Proof of Theorem 3.1 is provided in Section 3.5.1.

Moreover, we add a reconstruction loss to the loss function of E and G_{MI} to exploit the information of non-missing elements, as follows:

$$\mathcal{L}_{\mathrm{recon}} = \mathbb{E}_{\bar{\mathbf{x}}|\mathbf{x},\mathbf{m}} \left[\sum_{i=1}^{d} m_i (x_i - \bar{x}_i)^2 \right] \tag{3.5}$$

For more stable GAN training, we modify a simplified version of the zero-centered gradient penalty proposed by [104] in an element-wise manner, and add the gradient penalty to the loss function of D_{MI}. The modified regularizer penalizes the gradients of each output unit of the D_{MI} on $p_{\mathcal{D}}(x_i)$:

$$\mathcal{L}_{\mathrm{GP}_{MI}} = \sum_{i=1}^{d} \mathbb{E}_{p_{\mathcal{D}}(x_i)} \left[||\nabla_{\hat{\mathbf{x}}} D_{MI}(\hat{\mathbf{x}})_i||_2^2 \right] \tag{3.6}$$

We define $\hat{\mathbf{x}}$ in $p_{\mathcal{D}}(x_i)$ as data with m_i is 1 (i.e., $p_{\mathcal{D}}(x_i) = \{\hat{\mathbf{x}}^j | m_i^j = 1\}$). In other words, as suggested by [104], we penalize D_{MI} only for data wherein the i-th element is not missing (real) in a batch. This helps balance an adversarial relationship between the generator and discriminator by forcing the discriminator closer to Nash Equilibrium.

Therefore, missing data imputation and model training are performed as described in Algorithm 3.1. We used 10 for both hyperparameters λ_1 and α_1 in our experiments.

3.2.2 Conditional generation

We define conditional generation for the class imbalance problem as the imputation of entire data elements on a given class label (i.e., generating $(x_1, ..., x_d)$ following $p(\mathbf{x}|\mathbf{y})$). Since we have G_{MI}, which is a generator for imputation, we can oversample data instances by feeding synthetic $\mathbf{h}$ into G_{MI}. Therefore, we introduce G_{CG} to generate a hidden variable $\mathbf{h}_c$ corresponding to the target class label $\mathbf{y}_c$, i.e., we sample $\mathbf{h}_c \sim p_{G_{CG}}(\mathbf{h}|\mathbf{y})$. We also introduce D_{CG} to distinguish pairs of generated hidden variables and target class labels $(\mathbf{h}_c, \mathbf{y}_c)$ (fake) from pairs of hidden variables for labeled data and corresponding class labels $(\mathbf{h}_l, \mathbf{y}_l)$ (real). G_{CG} and D_{CG} are trained with WGAN loss and zero-centered gradient penalty on $\mathbf{h}_l$ as follows:

$$\mathcal{L}_{G_{CG}} = - \mathbb{E}_{\mathbf{h}_c \sim p_{G_{CG}}(\mathbf{h}_c|\mathbf{y}_c)}[D_{CG}(\mathbf{h}_c, \mathbf{y}_c)] \tag{3.7}$$

$$\begin{aligned}\mathcal{L}_{D_{CG}} =& \mathbb{E}_{\mathbf{h}_c \sim p_{G_{CG}}(\mathbf{h}_c|\mathbf{y}_c)}[D_{CG}(\mathbf{h}_c, \mathbf{y}_c)] \\ & - \mathbb{E}_{\mathbf{h}_l \sim p_E(\mathbf{h}_l|x_l)}[D_{CG}(\mathbf{h}_l, \mathbf{y}_l)]\end{aligned} \tag{3.8}$$

$$\mathcal{L}_{\mathrm{GP}_{CG}} = \mathbb{E}_{\mathbf{h}_l \sim p_E(\mathbf{h}_l|x_l)}\left[||\nabla_{\mathbf{h}_l} D_{CG}(\mathbf{h}_l, \mathbf{y}_l)||_2^2\right] \tag{3.9}$$

G_{MI} maps generated $\mathbf{h}_c$ into a realistic $\hat{\mathbf{x}}_c$. Because $\mathcal{L}_{G_{CG}}$ is not enough to stably generate $\mathbf{h}_c$, we add the loss of G_{MI} from $\hat{\mathbf{x}}_c$. Since we defined conditional generation as imputation of all the elements, G_{CG} and D_{MI} are related adversarially. The label of $(\hat{\mathbf{x}}_c, \mathbf{y}_c)$ for D_{MI} is a $(d + 1)$-dimensional zero vector.

In addition, the cross-entropy of $(\hat{\mathbf{x}}_c, \mathbf{y}_c)$ calculated from the prediction of C is also added to the loss function of G_{CG} to stably generate the data that is conditioned on the target class as follows:

$$\mathcal{L}_{\mathrm{CE}}(\hat{\mathbf{x}}_c, \mathbf{y}_c) = -\mathbb{E}_{\hat{\mathbf{x}}_c|\mathbf{y}_c}\left[\sum_{k=1}^{n_c} \mathbf{y}_{c_k} \log(C(\hat{\mathbf{x}}_c)_k)\right] \tag{3.10}$$

where $C(\cdot)_k$ is the softmax output for the k-th class. Thus, D_{CG} and G_{CG} are trained according to:

$$\min_{D_{CG}} \mathcal{L}_{D_{CG}} + \lambda_2 \mathcal{L}_{\mathrm{GP}_{CG}} \tag{3.11}$$

$$\min_{G_{CG}} \mathcal{L}_{G_{CG}} + \alpha_2 \mathcal{L}_{G_{MI}} + \alpha_3 \mathcal{L}_{\mathrm{CE}}(\hat{\mathbf{x}}_c, y_c) \tag{3.12}$$

where λ_2, α_2, and α_3 denote hyperparameters, and we set λ_2 to 10, α_2 to 1, and α_3 to 0.01 in our experiments. Since the distribution of $\mathbf{h}_l$ moves according to the training of E, we set the number of update iterations of D_{CG} and G_{CG} per an update of E to 10, so that $\mathbf{h}_c$ follows the distribution of $\mathbf{h}_l$ well.

3.2.3 Semi-supervised classification

Pseudo-labeling

We define semi-supervised learning as imputing missing labels by the pseudo-labeling technique, TripleGAN [95]. Semi-supervised learning is achieved by the interaction of C and D_{MI}. C generates a pseudo-label $\mathbf{y}_u$ of an unlabeled instance $\hat{\mathbf{x}}_u$, i.e., $\mathbf{y}_u$ is sampled from the classifier distribution $p_C(\mathbf{y}|\mathbf{x})$. Then, the data-label pair ($\hat{\mathbf{x}}_u$, $\mathbf{y}_u$) enters D_{MI}. The last element of the D_{MI} output, $D_{MI}(\cdot)_{d+1}$, determines whether the label is real or fake. The label for pseudo-labeling is m_y. C and D_{MI} are trained according to the following loss functions:

$$\mathcal{L}_C = -\mathbb{E}_{\mathbf{y}_u|\hat{\mathbf{x}}_u \sim p_C}\left[D_{MI}(\hat{\mathbf{x}}_u, \mathbf{y}_u)_{d+1}\right] \tag{3.13}$$

$$\mathcal{L}_{D_{MI}}^{d+1} = \mathbb{E}_{\mathbf{y}_u|\hat{\mathbf{x}}_u \sim p_C}\left[D_{MI}(\hat{\mathbf{x}}_u, \mathbf{y}_u)_{d+1}\right] \tag{3.14}$$

$$- \mathbb{E}_{\mathbf{y}|\hat{\mathbf{x}} \sim p_{data}} \left[D_{MI}(\hat{\mathbf{x}}, \mathbf{y})_{d+1} \right]$$

where p_{data} denotes the data distribution of y conditioned on $\hat{\mathbf{x}}$. $\mathcal{L}_{D_{MI}}^{d+1}$ is added to the loss of D_{MI}, so that i in Equation 3.4 expands from d to $d+1$. If G_{MI} learns the true data distribution, then we can postulate that p_{data} follows the true conditional distribution. We should note that the adversarial loss is identical to the loss function of WGAN between C and D_{MI}. Therefore, C plays a role as a label generator, and $D_{MI}(\cdot)_{d+1}$ acts as a label discriminator.

Through adversarial learning, we expect that the adversarial loss enhances the performance of C. The adversarial learning aims to minimize Earth Mover distance $W(\text{Distr}[C(\hat{\mathbf{x}}_u)], \text{Distr}[\mathbf{y}])$ by making C implicitly estimate the distributions of labels. Therefore, it can be said that C minimizing the adversarial loss $\mathcal{L}_C$ approximately optimizes the output distribution matching (ODM) cost [148].

According to the properties of the ODM cost, the global optimum of supervised learning is also a global optimum of semi-supervised learning. Therefore, intuitively, $\mathcal{L}_C$ and $\mathcal{L}_{D_{MI}}^{d+1}$ serve as guides for finding the optimum point of the supervised loss.

Classification of HexaGAN

In order to train C, the two models E and G_{MI} impute the missing values of data instances $\hat{\mathbf{x}}_l$. G_{CG} produces hidden vectors $\mathbf{h}_c$ conditioned on the minority classes so that the number of data in the minority classes in each batch is equal to the number of data instances in the majority class of each batch, and G_{MI} generates class-conditional data $\hat{\mathbf{x}}_c$. Then, the cross-entropy between $\hat{\mathbf{x}}_{l,c} \in (\hat{\mathbf{x}}_l \cup \hat{\mathbf{x}}_c)$ and $\mathbf{y}_{l,c} \in (\mathbf{y}_l \cup \mathbf{y}_c)$ is calculated to train C. Unlabeled data $\hat{\mathbf{x}}_u$ is used to optimize L_C, the loss for pseudo-labeling, thereby training a more robust classifier.

Therefore, C is trained according to:

$$\min_C \mathcal{L}_{\text{CE}}(\hat{\mathbf{x}}_{l,c}, \mathbf{y}_{l,c}) + \alpha_4 \mathcal{L}_C \tag{3.15}$$

Table 3.1: Dataset description. The imbalance ratio indicates the ratio of the number of instances in the majority class to the number of instances in the minority class.

Dataset	# of features	# of instances	Imbalance ratio (1:x)
Breast	30	569	1.68
Credit	23	30,000	3.52
Wine (with binarized class)	13	178	2.02
Madelon	500	4,400	1.00

where we used 0.1 for α_4 in our experiments. The entire training procedure of HexaGAN is presented in Section 3.5.2.

3.3 Experimental Setup

3.3.1 Datasets

Here, we present the performance of the proposed method. We used datasets from the UCI machine learning repository [36], including real world datasets (breast, credit, wine) and a synthetic dataset (madelon). Table 3.1 presents the dataset descriptions used in the experiments. The imbalance ratio of the wine dataset is calculated from the binarized classes by combining classes 2 and 3 into one class, and the numbers of data in the three classes are 59, 71, and 48, respectively. We also used a handwritten digit dataset (MNIST) [93].

3.3.2 Evaluation metrics

We basically assume 20% missingness (MCAR) in the elements and labels of the UCI dataset and 50% in the elements of the MNIST dataset to cause missing data and missing label problems. Every element was scaled to a range of $[0, 1]$. We repeated each experiment 10 times and used 5-fold cross validation. As the performance metric, we calculated the root mean square error (RMSE) for missing data imputation and the F1-score for classification. The implementation details are described in Section 3.5.2.

Table 3.2: Performance comparison with other imputation methods (RMSE)

Method	Breast	Credit	Wine	Madelon	MNIST
Zeros	0.2699	0.2283	0.4213	0.5156	0.3319
Matrix	0.0976	0.1277	0.1772	0.1456	0.2540
K-NN	0.0872	0.1128	0.1695	0.1530	0.2267
MICE	0.0842	0.1073	0.1708	0.1479	0.2576
Autoencoder	0.0875	0.1073	0.1481	0.1426	0.1506
GAIN	0.0878	0.1059	0.1406	0.1426	0.1481
HexaGAN	**0.0769**	**0.1022**	**0.1372**	**0.1418**	**0.1452**

3.4 Experimental Results

First, we show the imputation performance of HexaGAN. Then, we show the quality of conditional generation using our framework. Finally, we present the classification performance and ablation study of our proposed model, assuming the problems in real world classification.

3.4.1 Imputation performance

Comparison with real world datasets

We used UCI datasets and the MNIST dataset to evaluate the imputation performance. Table 3.2 shows the imputation performance of zero imputation, matrix completion, k-nearest neighbors, MICE, autoencoder, GAIN, and HexaGAN. In our experiments, we observed that HexaGAN outperforms the state-of-the-art methods on all datasets (up to a 14% improvement). Two deep generative models, GAIN and HexaGAN, use both the reconstruction loss and the adversarial loss. GAIN shows the same or lower performance than the autoencoder on certain datasets, whereas HexaGAN consistently outperforms the autoencoder on all datasets. This shows that the novel adversarial loss boosts the imputation performance.

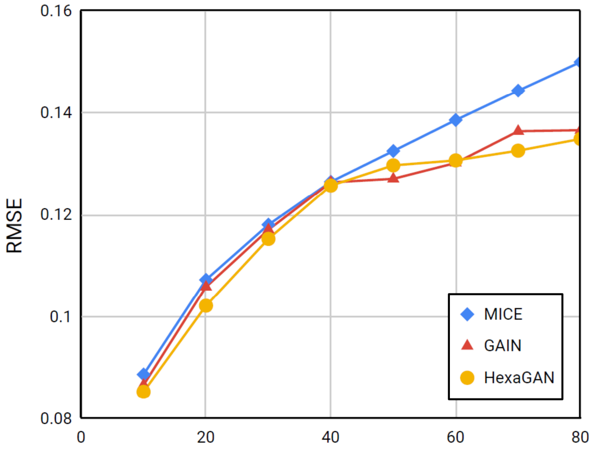


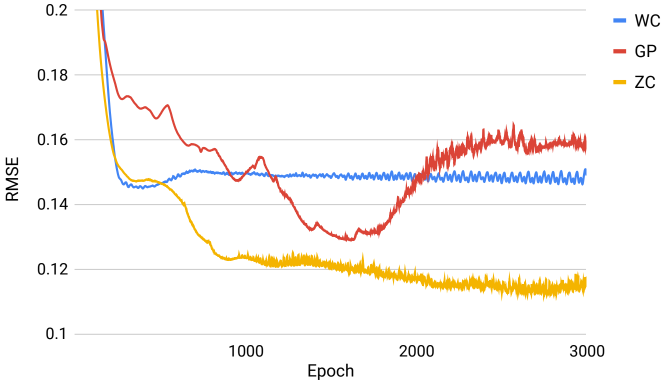
Figure 3.3: Imputation performance (RMSE) comparison with respect to the missing rate with the credit dataset

Imputation performance with respect to the missing rate

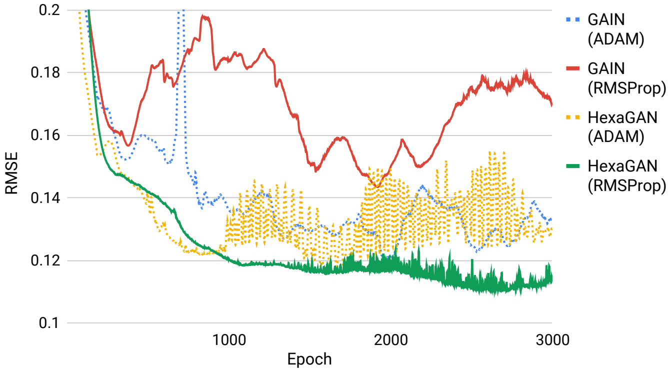
We measured the imputation performance of HexaGAN for various missing rates in the credit dataset. To compare the performance with those of competitive benchmarks, we used MICE, which is a state-of-the-art machine learning algorithm, and GAIN, which is a state-of-the-art deep generative model. As seen in Figure 3.3, HexaGAN shows the best performance for all missing rates except 50%. The comparison of MICE and HexaGAN shows that the gap between the performances of the two methods increases at higher missing rates; therefore, HexaGAN is more robust when there is less information available.

Learning curve analysis on missing data imputation

Using the breast dataset, we measured the RMSE to evaluate the imputation performance of the proposed adversarial losses ($\mathcal{L}_{D_{MI}}$, $\mathcal{L}_{G_{MI}}$). We excluded $\mathcal{L}_{\text{recon}}$ from the losses of E and G_{MI} and compared the learning curves of weight clipping (WC) [8], the modified gradient penalty (GP) [56], and the modified zero-centered gradient penalty (ZC, ours) to determine the most appropriate gradient penalty for our framework. As shown in Figure 3.4(a), ZC shows stable and good performance (small RMSE). In



(a) Comparison of the gradient penalty



(b) Comparison of the adversarial loss and optimizer

Figure 3.4: Learning curve comparison for the optimal GAN imputation method

Figure 3.4(b), we plot learning curves to accurately compare the adversarial losses of GAIN and HexaGAN. We also compare the two optimizers ADAM [82] and RMSProp [155]. Our experiment shows that RMSProp is a more stable optimizer than ADAM, and HexaGAN produces a more stable and better imputation performance than GAIN.

Qualitative analysis

Figure 3.5 visualizes the imputation performance with the MNIST dataset. Since MNIST is an image dataset, we designed HexaGAN with convolutional and deconvolutional neural networks. The first row of Figure 3.5 shows MNIST data with 50% missing

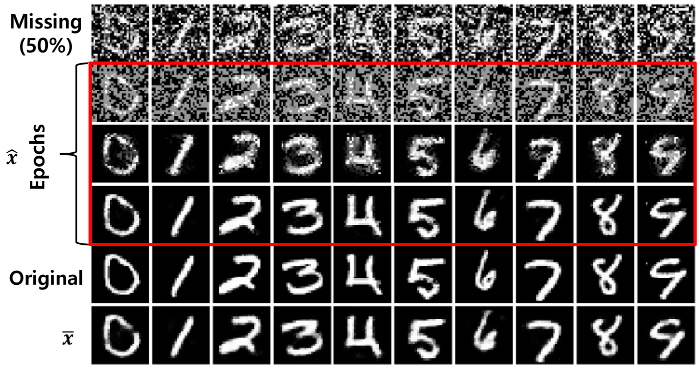


Figure 3.5: Imputation results with the MNIST dataset. 1st row: MNIST images with 50% missing randomly as inputs of HexaGAN. 2nd∼4th rows (red box): images imputed by HexaGAN ($\hat{\mathbf{x}}$) at 1, 10, and 100 epochs. 5th row: original images (no missing element). 6th row: images generated by G_{MI} for imputation ($\bar{\mathbf{x}}$).

as the input for HexaGAN. The next three rows show $\hat{\mathbf{x}}$ after 1, 10, and 100 epochs, and it can be seen that higher quality imputed data are generated as the number of epochs increases. The next row presents the original data with no missing value, and the last row shows $\bar{\mathbf{x}}$ generated by G_{MI}. This suggests that the proposed method imputes missing values with very high-quality data. The RMSE value using the convolutional architecture is 0.0914.

3.4.2 Conditional generation performance

TSNE analysis

We used tSNE [102] to analyze $\mathbf{h}_l$ generated by E and $\mathbf{h}_c$ generated by G_{CG}. Figure 3.6 shows the changes of $\mathbf{h}_l$ (circle) and $\mathbf{h}_c$ (triangle) according to the iteration. Each color stands for a class label. At epoch 1, $\mathbf{h}_l$ and $\mathbf{h}_c$ have very different distributions, and form respective clusters. At epoch 10, the cluster of $\mathbf{h}_c$ is overlapped by the cluster of $\mathbf{h}_l$. At epoch 100, E learns the manifold of the hidden representation, so that $\mathbf{h}_l$ is gathered by class and $\mathbf{h}_c$ follows the distribution of $\mathbf{h}_l$ well. That is, G_{CG} creates a high-quality $\mathbf{h}_c$ that is conditioned on a class label. The tSNE plot below shows an analysis of the manifold of the hidden space. We confirm that the synthetic data around

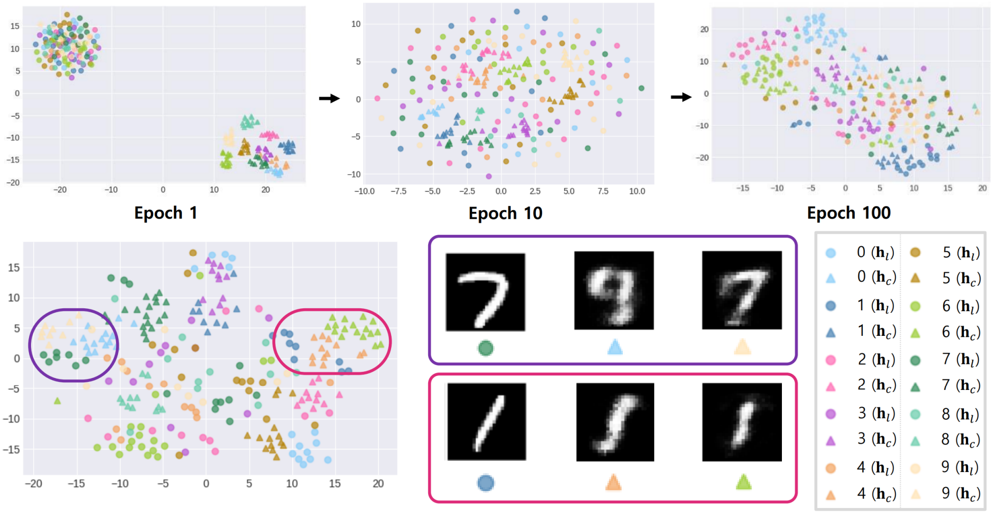


Figure 3.6: tSNE analysis with the MNIST dataset

the original data looks similar to the original data. Therefore, it can be seen that E learns the data manifold well in the hidden space.

Qualitative analysis

To evaluate the performance of conditional generation, we used the same architecture as in Section 3.4.1 and generated synthetic MNIST images conditioned on 10 class labels. Figure 3.7 presents the generated MNIST images. Each row shows the results of conditioning the class labels $0 \sim 9$, and each column shows the results of changing the noise vector $\mathbf{z}$. It can be seen that G_{CG} and G_{MI} produce realistic images of digits and that various image shapes are generated according to $\mathbf{z}$. Images conditioned on 9 in the second and fifth rows look like 7. This can be interpreted as a phenomenon in which the hidden variables for 9 and 7 are placed in adjacent areas on the manifold of the hidden space.

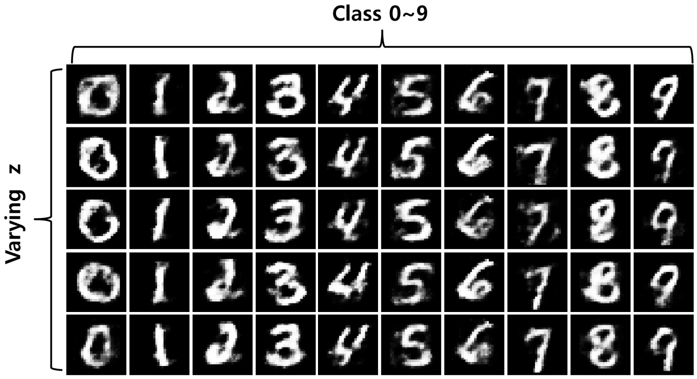


Figure 3.7: Class-conditional generation results with the MNIST dataset. Each row visualizes generated images conditioned on 0∼9. Each column shows images generated by all different **z**s.

3.4.3 Classification performance

HexaGAN works without any problem for multi-class classification, but for the convenience of the report, we tested only binary classifications. The breast and credit datasets are imbalanced with a large number of negative samples. The wine dataset has three classes, and it was tested by binarizing the label 1 as negative, and labels 2 and 3 as positive to calculate an F1-score. The wine dataset was imbalanced with a large number of positive samples. Madelon is a balanced synthetic dataset that randomly assigns binary labels to 32 clusters on 32 vertices of a 5-dimensional hypercube.

Comparison with other combinations

In this experiment, we compared the classification performance of HexaGAN with those of combinations of state-of-the-art methods for the three problems. For missing data imputation, we used MICE, which showed the best performance among machine learning based methods, and GAIN, which showed the best performance among deep generative models. For class imbalance, we used the cost sensitive loss (CS) and oversampled the minority class in a batch using SMOTE. We adopted the TripleGAN for semi-supervised learning. The classifier of TripleGAN used the same architecture

Table 3.3: Classification performance (F1-score) comparison with other combinations of state-of-the-art methods

Method	Breast	Credit	Wine	Madelon
MICE + CS + TripleGAN	0.9417 ± 0.0044	0.3836 ± 0.0052	0.9704 ± 0.0043	0.6681 ± 0.0028
GAIN + CS + TripleGAN	0.9684 ± 0.0102	0.4076 ± 0.0038	0.9727 ± 0.0046	0.6690 ± 0.0027
MICE + SMOTE + TripleGAN	0.9434 ± 0.0060	0.4163 ± 0.0029	0.9756 ± 0.0037	0.6712 ± 0.0008
GAIN + SMOTE + TripleGAN	0.9672 ± 0.0063	0.4401 ± 0.0031	0.9735 ± 0.0063	0.6703 ± 0.0032
HexaGAN	**0.9762 ± 0.0021**	**0.4627 ± 0.0040**	**0.9814 ± 0.0059**	**0.6716 ± 0.0019**

as C of HexaGAN for a fair comparison.

As shown in Table 3.3, HexaGAN shows significantly better performance than the combinations of existing methods in cascading form (up to a 5% improvement). In addition, the madelon dataset is balanced; thus, comparing HexaGAN without G_{CG} (the third row of Table 4.4) with the combination of MICE, CS, and TripleGAN (the first row of Table 3.3) and the combination of GAIN, CS, and TripleGAN (the second row of Table 3.3) shows the classification performance with respect to imputation methods. We confirm that the imputation method of HexaGAN guarantees better classification performance than the other imputation methods.

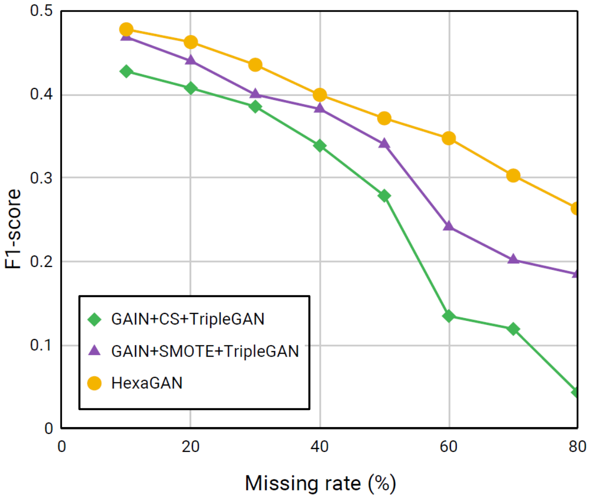


Figure 3.8: Classification performance (F1-score) comparison with respect to the missing rate with the credit dataset

Classification performance with respect to missing rate

Figure 3.8 compares the classification performance of HexaGAN with those of competitive combinations for various missing rates in the credit dataset. We used the combination of GAIN, CS, and TripleGAN and the combination of GAIN, SMOTE, and TripleGAN as benchmarks. According to the results, HexaGAN outperforms the benchmarks for all missing rates. Moreover, our method shows a larger performance gap compared to the benchmarks for high missing rates. This means that HexaGAN works robustly in situations in which only little information is available.

Classification performance with the CelebA dataset

We used a more challenging dataset, CelebA. It is a high-resolution face dataset for which it is more difficult to impute missing data. CelebA consists of 40 binary attributes with various imbalance ratios (1:1 $\sim$ 1:43). We used 50,000 and 10,000 labeled and unlabeled training images, respectively, and 10,000 test images. The size of each image is 218x178x3, which means that the data dimension is 116,412. Therefore, we could evaluate our method on the setting where the data dimension is less than the sample size. Then, half of the elements were removed from each image under the 50% missingness (MCAR) assumption.

For comparison, we utilized a class rectification loss (CRL) [38] which is a recent method developed for the class imbalance problem. Since an image has 40 labels simultaneously, we simply balanced the class of data entered into C by setting the class condition to $\mathbf{1} - \mathbf{y}$. Additionally, the data dimension was too large to calculate $\mathcal{L}_{\mathrm{GP}_{MI}}$, therefore we replaced the regularization for discriminator learning with weight clipping. We measured the F1-scores for 40 attributes for three cases: GAIN + TripleGAN, GAIN + CRL + TripleGAN, and HexaGAN. The same structure and hyperparameters were used for the classifier for a fair comparison. Table 3.4 shows the imbalance ratio of each attribute and the classification performance (F1-score) of each combination. Comparing the average F1-score of 40 attributes, GAIN + TripleGAN shows a performance of

Table 3.4: Classification performance comparison with the CelebA dataset (F1-score)

Attribute	Imb. ratio (1:x)	GAIN + TripleGAN	GAIN + CRL + TripleGAN	HexaGAN
Arched eyebrows	3	0.53	0.50	**0.55**
Attractive	1	**0.78**	0.74	0.74
Bags under eyes	4	0.30	0.44	**0.49**
Bald	43	0.37	**0.42**	0.35
Bangs	6	0.70	**0.77**	0.71
Big lips	3	0.17	0.20	**0.39**
Big nose	3	0.41	0.47	**0.49**
Black hair	3	0.67	**0.72**	0.69
Blond hair	6	**0.77**	0.74	0.71
Blurry	18	0.02	**0.16**	0.15
Brown hair	4	0.49	0.49	**0.57**
Bushy eyebrows	6	0.48	**0.55**	0.49
Chubby	16	**0.49**	0.33	0.45
Double chin	20	0.34	0.36	**0.46**
Eyeglasses	14	0.64	**0.81**	0.79
Goatee	15	0.41	0.48	**0.50**
Gray hair	23	0.46	0.55	**0.59**
Heavy makeup	2	0.80	**0.84**	**0.84**
High cheekbones	1	0.78	0.79	**0.80**
Male	1	0.91	**0.93**	**0.93**
Mouth slightly open	1	0.81	**0.83**	0.82
Mustache	24	0.36	**0.58**	0.49
Narrow eyes	8	0.17	0.25	**0.28**
No beard	5	0.95	**0.95**	0.92
Oval face	3	0.16	0.24	**0.47**
Pale skin	22	0.34	**0.45**	0.39
Pointy nose	3	0.49	0.31	**0.52**
Receding hairline	11	0.22	**0.46**	0.44
Rosy cheeks	14	0.45	0.53	**0.55**
Shadow	8	0.45	**0.49**	0.46
Sideburns	17	0.50	0.58	**0.60**
Smiling	1	0.85	**0.87**	**0.87**
Straight hair	4	0.30	0.07	**0.38**
Wavy hair	2	0.52	0.50	**0.57**
Wearing earrings	4	0.44	0.48	**0.53**
Wearing hat	19	0.65	0.67	**0.70**
Wearing lipstick	1	**0.88**	**0.88**	**0.88**
Wearing necklace	7	0.04	0.11	**0.35**
Wearing necktie	13	0.62	**0.65**	0.63
Young	4	**0.89**	**0.89**	0.76
Mean	-	0.5152	0.5519	**0.5826**

0.5152, GAIN + CRL + TripleGAN has a performance of 0.5519, and HexaGAN has a performance of 0.5826. HexaGAN outperforms all the compared methods.

3.4.4 Ablation study

Component analysis

The components affecting the classification performance of HexaGAN are G_{MI} to fill in missing data, G_{CG} to perform conditional generation, and $D_{MI}(\cdot)_{d+1}$ to enable semi-supervised learning. Table 3.5 compares the classification performance depending on the removal of these components. In the case of MLP, which is equivalent to HexaGAN without any of these three components, missing data were filled in with values sampled uniformly from [0,1].

As a result, MLP shows the worst performance. When HexaGAN contains G_{CG} (from the second row to the fourth row), the biggest performance improvement is shown in the credit data which is the most imbalanced. The more components included in HexaGAN, the higher the classification performance obtained. HexaGAN with all components shows the highest performance on every dataset. Our delicately devised architecture improves the classification performance by up to 36%. It offers the advantage that any classifier that is state-of-the-art in a controlled environment can be plugged into the proposed framework, and the classifier will perform at its highest capacity.

In addition, we conducted an ablation study to evaluate whether HexaGAN improves classification performance when the missing rate is 0. Table 3.6 indicates that even in cases where there is no missing data, HexaGAN learns the information of the data via reconstruction loss and improves performance through oversampling and semi-supervised learning.

Sensitivity analysis of loss functions

We performed diverse experiments by tuning the hyperparameter of each loss term for the missing data imputation and conditional generation experiments. We utilized the

Table 3.5: Ablation study of HexaGAN (F1-score)

Method	Breast	Credit	Wine	Madelon
MLP (HexaGAN w/o G_{MI} & G_{CG} & $D_{MI_{d+1}}$)	0.9171 ± 0.0101	0.3404 ± 0.0080	0.9368 ± 0.0040	0.6619 ± 0.0017
HexaGAN w/o G_{CG} & $D_{MI_{d+1}}$	0.9725 ± 0.0042	0.4312 ± 0.0028	0.9724 ± 0.0065	0.6676 ± 0.0038
HexaGAN w/o G_{CG}	0.9729 ± 0.0007	0.4382 ± 0.0075	0.9738 ± 0.0135	0.6695 ± 0.0043
HexaGAN w/o $D_{MI_{d+1}}$	0.9750 ± 0.0030	0.4604 ± 0.0097	0.9770 ± 0.0037	0.6699 ± 0.0022
HexaGAN	**0.9762** ± **0.0021**	**0.4627** ± **0.0040**	**0.9814** ± **0.0059**	**0.6716** ± **0.0019**

Table 3.6: Ablation study when missing rate is 0

Method	Breast	Credit	Wine	Madelon
MLP	0.9764 ± 0.0013	0.4711 ± 0.0029	0.9954 ± 0.0002	0.6795 ± 0.0024
HexaGAN	**0.9853** ± **0.0008**	**0.4943** ± **0.0028**	**0.9965** ± **0.0020**	**0.6832** ± **0.0024**

Table 3.7: Sensitivity analysis of the loss functions with the credit dataset

Hyperparameter (Loss)	Setting	1	2	3	4
α_1 ($\mathcal{L}_{\text{recon}}$)	Value	0	1	**10**	100
	RMSE	0.1974	0.1108	**0.1022**	0.1079
λ_1 ($\mathcal{L}_{\text{GP}_{MI}}$)	Value	0	1	**10**	100
	RMSE	0.1110	0.1097	**0.1022**	0.1081
α_2 ($\mathcal{L}_{G_{MI}}$)	Value	0	**1**	10	100
	F1-score	0.4535	**0.4627**	0.4585	0.4523
α_3 ($\mathcal{L}_{\text{CE}}(\hat{\mathbf{x}}_c, \mathbf{y}_c)$)	Value	0	**0.01**	0.1	1
	F1-score	0.4535	**0.4627**	0.4585	0.4523

credit dataset and measured the RMSE and F1-score. The first two rows of Table 3.7 show the imputation performances (RMSE) acheived by tuning hyperparameters α_1 and λ_1, which are multiplied by the auxiliary loss terms for missing data imputation ($\mathcal{L}_{\text{recon}}$ and $\mathcal{L}_{\text{GP}_{MI}}$, respectively). The results show that HexaGAN achieves the best missing data imputation performance when both α_1 and λ_1 are set to 10. The last two rows of Table 3.7 present the classification performances (F-score) acheived by tuning hyperparameters α_2 and α_3, which are multiplied by the auxiliary losses for conditional generation ($\mathcal{L}_{G_{MI}}$ and $\mathcal{L}_{\text{CE}}(\hat{\mathbf{x}}_c, \mathbf{y}_c)$, respectively). As a result, the best classification performance is obtained when α_2 and α_3 are the default values in our paper, at 1 and 0.01, respectively.

3.5 Appendix

3.5.1 Proofs

Global optimality of $p(\mathbf{x}|\mathbf{m}_i = 1) = p(\mathbf{x}|\mathbf{m}_i = 0)$ for HexaGAN

Proof of Theorem 3.1: Let $D_{MI}(\cdot)$ be $D(\cdot)$, and $G_{MI}(E(\cdot))$ be $G(\cdot)$ for convenience.

The min-max loss of HexaGAN for missing data imputation is given by:

$$V_{MI}(D, G) = \mathbb{E}_{\mathbf{x},\mathbf{z},\mathbf{m}}\left[\mathbf{m}^T D(G(\tilde{\mathbf{x}}|\mathbf{m})) - (\mathbf{1} - \mathbf{m})^T D(G(\tilde{\mathbf{x}}|\mathbf{m}))\right] \tag{3.16}$$

$$= \mathbb{E}_{\hat{\mathbf{x}},\mathbf{m}}\left[\mathbf{m}^T D(\hat{\mathbf{x}}) - (\mathbf{1} - \mathbf{m})^T D(\hat{\mathbf{x}})\right] \tag{3.17}$$

$$= \int_{\hat{\mathcal{X}}} \sum_{\mathbf{m}\in\{0,1\}^d} \left(\mathbf{m}^T D(\mathbf{x}) - (1 - \mathbf{m})^T D(\mathbf{x})\right) p(\mathbf{x}|\mathbf{m}) d\mathbf{x} \tag{3.18}$$

$$= \int_{\hat{\mathcal{X}}} \sum_{\mathbf{m}\in\{0,1\}^d} \left(\sum_{i:m_i=1} D(\mathbf{x})_i - \sum_{i:m_i=0} D(\mathbf{x})_i\right) p(\mathbf{x}|\mathbf{m}) d\mathbf{x} \tag{3.19}$$

$$= \int_{\hat{\mathcal{X}}} \sum_{i=1}^{d} \left(D(\mathbf{x})_i \sum_{\mathbf{m}:m_i=1} p(\mathbf{x}|\mathbf{m}) - D(\mathbf{x})_i \sum_{\mathbf{m}:m_i=0} p(\mathbf{x}|\mathbf{m})\right) d\mathbf{x} \tag{3.20}$$

$$= \int_{\hat{\mathcal{X}}} \sum_{i=1}^{d} D(\mathbf{x})_i p(\mathbf{x}|m_i = 1) - D(\mathbf{x})_i p(\mathbf{x}|m_i = 0) d\mathbf{x} \tag{3.21}$$

$$= \int_{\hat{\mathcal{X}}} \sum_{i=1}^{d} \left(p(\mathbf{x}|m_i = 1) - p(\mathbf{x}|m_i = 0)\right) D(\mathbf{x})_i d\mathbf{x} \tag{3.22}$$

For a fixed G, the optimal discriminator $D(\mathbf{x})_i$ which maximizes $V_{MI}(D, G)$ is such that:

$$D^*_G(\mathbf{x})_i = \begin{cases} 1, & \text{if } p(\mathbf{x}|m_i = 1) \geq p(\mathbf{x}|m_i = 0) \\ 0, & \text{otherwise} \end{cases} \tag{3.23}$$

Plugging D_G^* back into Equation 3.22, we get:

$$V_{MI}(D_G^*, G) = \int_{\hat{\mathcal{X}}} \sum_{i=1}^{d} \left(p(\mathbf{x}|m_i = 1) - p(\mathbf{x}|m_i = 0)\right) D_G^*(\mathbf{x})_i d\mathbf{x} \tag{3.24}$$

$$= \sum_{i=1}^{d} \int_{\{\mathbf{x}|p(\mathbf{x}|m_i=1) \geq p(\mathbf{x}|m_i=0)\}} \left(p(\mathbf{x}|m_i = 1) - p(\mathbf{x}|m_i = 0)\right) d\mathbf{x} \tag{3.25}$$

Let $\mathcal{X} = \{\mathbf{x}|p(\mathbf{x}|m_i = 1) \geq p(\mathbf{x}|m_i = 0)\}$. To minimize Equation 3.25, we need to set $p(\mathbf{x}|m_i = 1) = p(\mathbf{x}|m_i = 0)$ for $\mathbf{x} \in \mathcal{X}$.

Then, when we consider $\mathcal{X}^c$, the complement of $\mathcal{X}$, $p(\mathbf{x}|m_i = 1) < p(\mathbf{x}|m_i = 0)$ for $\mathbf{x} \in \mathcal{X}^c$. Since both probability density functions should integrate to 1,

$$\int_{\mathcal{X}^c} p(\mathbf{x}|m_i = 1) d\mathbf{x} = \int_{\mathcal{X}^c} p(\mathbf{x}|m_i = 0) d\mathbf{x} \tag{3.26}$$

However, this is a contradiction, unless $\lambda(X^c) = 0$ where λ is the Lebesgue measure. This finishes the proof. □

Optimization of components for imputation

From Equation 3.21,

$$V_{MI}(D, G)_i = \int_{\hat{\mathcal{X}}} p(\mathbf{x}|m_i = 1) D(\mathbf{x})_i - p(\mathbf{x}|m_i = 0) D(\mathbf{x})_i d\mathbf{x} \tag{3.27}$$

$$= \mathbb{E}_{\tilde{\mathbf{x}},\mathbf{z},\mathbf{m}} \left[m_i \cdot D(G(\tilde{\mathbf{x}}|\mathbf{m}))_i\right] - \mathbb{E}_{\tilde{\mathbf{x}},\mathbf{z},\mathbf{m}} \left[(1 - m_i) \cdot D(G(\tilde{\mathbf{x}}|\mathbf{m}))_i\right] \tag{3.28}$$

G is then trained according to $\min_G \sum_{i=1}^{d} V_{MI}(D, G)_i$, and D is trained according to $\max_D \sum_{i=1}^{d} V_{MI}(D, G)_i$.

Algorithm 3.2 Training procedure of HexaGAN

```
Require: $n_{CG}$ - the number of iterations for the conditional generation per iteration for the other components;
         $n_{critic}$ - the number of iterations for discriminators per iteration for generators
  while training loss is not converged do
    (1) Missing data imputation
    for $k = 1, ..., n_{critic}$ do
      Update $D_{MI}$ using stochastic gradient descent (SGD)
      $\nabla_{D_{MI}} \mathcal{L}_{D_{MI}} + \mathcal{L}^{d+1}_{D_{MI}} + \lambda_1 \mathcal{L}_{\mathrm{GP}_{MI}}$
    end for
    Update $E$ using SGD
    $\nabla_E \mathcal{L}_{G_{MI}} + \alpha_1 \mathcal{L}_{\mathrm{recon}}$
    Update $G_{MI}$ using SGD
    $\nabla_{G_{MI}} \mathcal{L}_{G_{MI}} + \alpha_1 \mathcal{L}_{\mathrm{recon}}$

    (2) Conditional generation
    for $i = 1, ..., n_{CG}$ do
      for $j = 1, ..., n_{critic}$ do
        Update $D_{CG}$ using SGD
        $\nabla_{D_{CG}} \mathcal{L}_{D_{CG}} + \lambda_2 \mathcal{L}_{\mathrm{GP}_{CG}}$
      end for
      Update $G_{CG}$ using SGD
      $\nabla_{G_{CG}} \mathcal{L}_{G_{CG}} + \alpha_2 \mathcal{L}_{G_{MI}} + \alpha_3 \mathcal{L}_{\mathrm{CE}}(\hat{\mathbf{x}}_c, \mathbf{y}_c)$
    end for

    (3) Semi-supervised classification
    Update $C$ using SGD
    $\nabla_C \mathcal{L}_{\mathrm{CE}}(\hat{\mathbf{x}}_{l,c}, \mathbf{y}_{l,c}) + \alpha_4 \mathcal{L}_C$
  end while
```

3.5.2 Implementation details

Training procedure

Each component of the whole system is updated in order. We should note that the distribution of $\mathbf{h}_l$ is altered by the updating of E; thus, we updated G_{CD} and D_{CG} several times when the other components are updated once, as shown in Algorithm 3.2. We set the number of iterations for the conditional generation per an iteration for the other components to 10 and the number of iterations for discriminators per an iteration for generators to 5 in our experiments.

Architecture of HexaGAN

Excluding the experiments on the MNIST dataset, all six components used an architecture with three fully-connected layers. The number of hidden units in each layer is d, $d/2$, and d. As an activation function, we use the rectified linear unit (ReLU) function for all hidden layers and the output layer of E and G_{CG}, the sigmoid function for the output layer of G_{MI} and D_{CG}, no activation function for the output layer of D_{MI}, and the softmax function for the output layer of C.

Table 3.8 describes the network architectures for the MNIST dataset. In the table, FC(n) denotes a fully-connected layer with n output units. Conv(n, $k \times k$, s) denotes a convolutional network with n feature maps, filter size $k \times k$, and stride s. Deconv(n, $k \times k$, s) denotes a deconvolutional network with n feature maps, filter size $k \times k$, and stride s.

Code Availability

Code is available at https://github.com/shinyflight/hexagan

Table 3.8: Convolutional neural network architectures used for the MNIST dataset

G_{CG}	D_{CG}	E	G_{MI}	D_{MI}	C

FC(512)
ReLU
FC(1024)
ReLU
FC(2048)
ReLU
	FC(1024)
	ReLU
	FC(512)
	ReLU
	FC(1)
	Sigmoid
		Conv(32, 5×5, 2)
		ReLU
		Conv(64, 5×5, 2)
		ReLU
		Conv(128, 5×5, 2)
		ReLU
			Deconv(64, 5×5, 2)
			ReLU
			Deconv(32, 5×5, 2)
			ReLU
			Deconv(1, 5×5, 2)
			ReLU
			FC(784)
			Sigmoid
				Conv(32, 5×5, 2)
				ReLU
				Conv(64, 5×5, 2)
				ReLU
				Conv(128, 5×5, 2)
				ReLU
				FC(785)
				Sigmoid
					Conv(32, 5×5, 2)
					ReLU
					Conv(64, 5×5, 2)
					ReLU
					Conv(128, 5×5, 2)
					ReLU
					FC(10)
					Softmax

3.6 Summary

In this chapter, we interactively overcome the three main problems in real world classification (missing data, class imbalance, and missing label). We define the three problems from the perspective of missing information. Then, we propose a HexaGAN framework wherein six neural networks are actively correlated with others, and design several loss functions that maximize the utilization of any incomplete data.

Our proposed method encourages more powerful performance in both imputation and classification than existing state-of-the-art methods. Moreover, HexaGAN is a one-stop solution that automatically solves the three problems commonly presented in real world classification.

For future work, we plan to extend HexaGAN to time series datasets such as electronic health records.

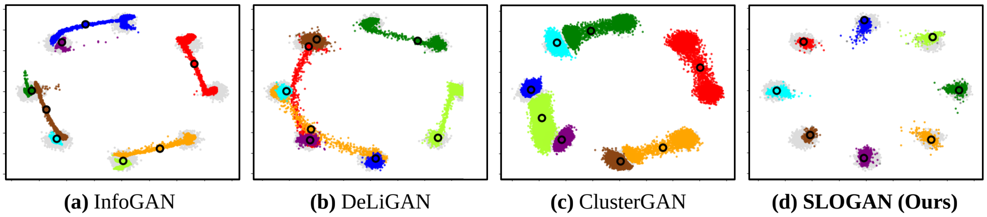
Chapter 4

GANs for Unsupervised Conditional Generation

4.1 Introduction

GANs have shown remarkable results in the synthesis of realistic data conditioned on a specific class [118, 111, 76]. Training conditional GANs requires a massive amount of labeled data; however, data are often unlabeled or possess only a few labels. For unsupervised conditional generation, the salient attributes of the data are first identified by unsupervised learning and used for conditional generation of data. Recently, several unsupervised conditional GANs have been proposed [27, 116, 120, 9]. By maximizing a lower bound of mutual information between latent codes and generated data, they cluster the attributes of the underlying data distribution in their latent spaces. These GANs achieve satisfactory performance when the salient attributes of data are balanced.

However, the attributes of real-world data can be *imbalanced*. For example, in the CelebA dataset [101], examples with one attribute (not wearing eyeglasses) outnumber the other attribute (wearing eyeglasses). Similarly, the number of examples with disease-related attributes in a biomedical dataset might be miniscule [70]. Thus, the imbalanced nature of real-world attributes must be considered for unsupervised conditional generation. Most of existing unsupervised conditional GANs are not suitable for real-world attributes, because they assume balanced attributes if the imbalance ratio is



(a) InfoGAN **(b)** DeLiGAN **(c)** ClusterGAN **(d) SLOGAN (Ours)**

Figure 4.1: Unsupervised conditional generation on synthetic dataset. Dataset consists of eight two-dimensional Gaussians (gray dots), and the number of unlabeled data instances from each Gaussian distribution is imbalanced (clockwise from the top, imbalance ratio between the first four Gaussians and the remaining four is 1:3). It is considered that the instances sampled from the same Gaussian share an attribute. Dots with different colors denote the data generated from different latent codes. Bold circles represent the samples generated from the mean vectors of latent distributions.

unknown [27, 116, 120]. Examples where existing methods fail to learn imbalanced attributes are shown in Figure 4.1 (a), (b) and (c).

In this chapter, we propose unsupervised conditional GANs, referred to as Stein Latent Optimization for GANs (SLOGAN). We define the latent distribution of the GAN models as Gaussian mixtures to enable the imbalanced attributes to be naturally clustered in a continuous latent space. We derive reparameterizable gradient identities for the mean vectors, full covariance matrices, and mixing coefficients of the latent distribution using Stein's lemma. This enables stable learning and makes latent distribution parameters, including the mixing coefficient, learnable. We then devise a GAN framework with an encoder network and an unsupervised conditional contrastive loss (U2C loss), which can interact well with the learnable Gaussian mixture prior (Figure 4.2). This framework facilitates the association of data generated from a Gaussian component with a single attribute.

For the synthetic dataset, our method (Figure 4.1 (d)) shows superior performance on unsupervised conditional generation, with the accurately learned mixing coefficients. We performed experiments on various real-world datasets including MNIST [93], Fashion-MNIST [169], CIFAR-10 [88], CelebA [101], CelebA-HQ [78], and AFHQ [29] using architectures such as DCGAN [129], ResGAN [56], and StyleGAN2 [80].

Through experiments, we verified that the proposed method outperforms existing unsupervised conditional GANs in unsupervised conditional generation on datasets with balanced or imbalanced attributes. Furthermore, we confirmed that we could control the attributes to be learned when a small set of probe data is provided.

The contributions are summarized as follows:

- We propose novel Stein Latent Optimization for GANs (SLOGAN). To the best of our knowledge, this is one of the first methods that can perform unsupervised conditional generation by considering the imbalanced attributes of real-world data.

- To enable this, we derive the implicit reparameterization for Gaussian mixture prior using Stein's lemma. Then, we devise a GAN framework with an encoder and an unsupervised conditional contrastive loss (U2C loss) suitable for implicit reparameterization.

- SLOGAN significantly outperforms the existing methods on unsupervised learning tasks, such as cluster assignment, unconditional data generation, and unsupervised conditional generation, on datasets that include balanced or imbalanced attributes.

4.2 Stein Latent Optimization for GANs

In the following subsections, we propose Stein Latent Optimization for GANs (SLOGAN). We assume a Gaussian mixture prior (Section 4.2.1), derive implicit reparameterization of the parameters of the mixture prior (Section 4.2.2), and construct a GAN framework with U2C loss (Section 4.2.3). Additionally, we devise a method to assign a cluster (Section 4.2.4) and manipulate attributes to be learned if necessary (Section 4.2.5). An overview of SLOGAN is shown in Figure 4.2.

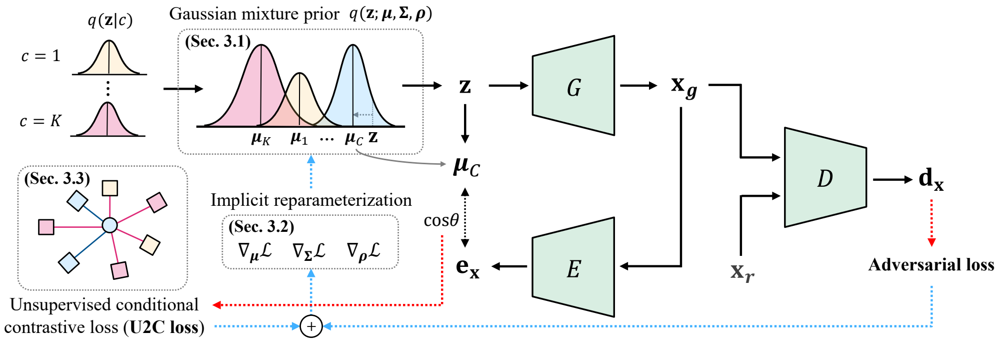

$q(\mathbf{z}|c)$
Gaussian mixture prior $q(\mathbf{z}; \boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\rho})$
(Sec. 3.1)
$c = 1$
$c = K$
$\boldsymbol{\mu}_K$ $\boldsymbol{\mu}_1$ ⋯ $\boldsymbol{\mu}_C$ $\mathbf{z}$
$\mathbf{z}$
G
$\mathbf{x}_g$
D
$\mathbf{d}_\mathbf{x}$
$\boldsymbol{\mu}_C$
$\cos\theta$
$\mathbf{e}_\mathbf{x}$
E
$\mathbf{x}_r$
Adversarial loss
(Sec. 3.3)
Implicit reparameterization
(Sec. 3.2)
$\nabla_{\boldsymbol{\mu}}\mathcal{L}$ $\nabla_{\boldsymbol{\Sigma}}\mathcal{L}$ $\nabla_{\boldsymbol{\rho}}\mathcal{L}$
Unsupervised conditional
contrastive loss (U2C loss)

Figure 4.2: Overview of the SLOGAN model. Here, $\mathbf{x}_g$ denotes the data generated from a latent vector $\mathbf{z}$, $\mathbf{x}_r$ is real data that is used for adversarial learning, and C indicates a component ID of the Gaussian mixture prior with the highest responsibility $\text{argmax}_c\, q(c|\mathbf{z})$.

4.2.1 Gaussian mixture prior

We consider a GAN with a generator $G : \mathbb{R}^{d_z} \mapsto \mathbb{R}^{d_x}$ and a discriminator $D : \mathbb{R}^{d_x} \mapsto \mathbb{R}$, where d_z and d_x are the dimensions of latent and data spaces, respectively. In the latent space $\mathcal{Z} \in \mathbb{R}^{d_z}$, we consider a conditional latent distribution $q(\mathbf{z}|c) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}_c, \Sigma_c)$, $c = 1, ..., K$, where K is the number of components we initially set and $\boldsymbol{\mu}_c$, Σ_c are the mean vector and covariance matrix of the c-th component, respectively. Subsequently, we consider a Gaussian mixture $q(\mathbf{z}) = \sum_{c=1}^{K} p(c)q(\mathbf{z}|c)$ parameterized by $\boldsymbol{\mu} = \{\boldsymbol{\mu}_c\}_{c=1}^{K}$, $\boldsymbol{\Sigma} = \{\Sigma_c\}_{c=1}^{K}$ and $\boldsymbol{\pi} = \{\pi_c\}_{c=1}^{K} = \{p(c)\}_{c=1}^{K}$ as the prior.

We hypothesize that a mixture prior in a continuous space could model some continuous attributes of real-world data (e.g., hair color) more naturally than categorical priors which could introduce discontinuity [116]. Because we use implicit reparameterization of a mixture of Gaussian priors (derived in Section 4.2.2), SLOGAN can fully benefit from implicit reparameterization and U2C loss. By contrast, the implicit reparameterization of prior distributions that do not belong to the exponential family (e.g., categorical priors) remains an open question.

In the experiments, the elements of $\boldsymbol{\mu}_c$ were sampled from $\mathcal{N}(0, 0.1)$, and we selected $\Sigma_c = I$ and $\pi_c = 1/K$ as the initial values. For the convenience of notation, we define the latent distribution $q = q(\mathbf{z})$, the mixing coefficient $\pi_c = p(c)$, and $\boldsymbol{\delta}(\mathbf{z}) = \{\delta(\mathbf{z})_c\}_{c=1}^{K}$, where $\delta(\mathbf{z})_c = q(\mathbf{z}|c)/q(\mathbf{z})$. $q(c|\mathbf{z})$, the responsibility of component c for a latent vector $\mathbf{z}$, can be expressed as follows:

$$q(c|\mathbf{z}) = \frac{q(c, \mathbf{z})}{q(\mathbf{z})} = \frac{q(\mathbf{z}|c)p(c)}{q(\mathbf{z})} = \delta(\mathbf{z})_c \pi_c \tag{4.1}$$

4.2.2 Gradient identities

We present gradient identities for the latent distribution parameters. To derive the identities, we use the generalized Stein's lemma for Gaussian mixtures with full covariance matrices [98]. First, we derive a gradient identity for the mean vector using Bonnet's theorem [17].

Theorem 4.1. *Given an expected loss of the generator $\mathcal{L}$ and a loss function for a sample $\ell(\cdot) : \mathbb{R}^{d_z} \mapsto \mathbb{R}$, we assume ℓ is locally absolute continuous on almost every straight line (ACL) and continuous. Then, the following identity holds:*

$$\nabla_{\boldsymbol{\mu}_c} \mathcal{L} = \mathbb{E}_q \left[\delta(\mathbf{z})_c \pi_c \nabla_{\mathbf{z}} \ell(\mathbf{z}) \right] \tag{4.2}$$

Proof of Theorem 4.1 is given in Section 4.5.1.

We derive a gradient identity for the covariance matrix via Price's theorem [128]. Among the two versions of the Price's theorem, we use the first-order identity to minimize computational cost.

Theorem 4.2. *Under the same assumptions as in Theorem 4.1 and assuming that $\mathbb{E}_q[\ell(\mathbf{z})]$ is well-defined, the following gradient identity holds:*

$$\nabla_{\Sigma_c} \mathcal{L} = \frac{1}{2} \mathbb{E}_q \left[\delta(\mathbf{z})_c \pi_c \Sigma_c^{-1} \left(\mathbf{z} - \boldsymbol{\mu}_c \right) \nabla_{\mathbf{z}}^T \ell(\mathbf{z}) \right] \tag{4.3}$$

Proof of Theorem 4.2 is given in Section 4.5.1. In the implementation, we replaced the expectation of the right-hand side of Equation 4.3 with the average for a batch of latent vectors; hence, the updated Σ_c may not be symmetric or positive-definite. To force a valid covariance matrix, we modify the updates of the covariance matrix as follows:

$$\Delta\Sigma_c = -\nabla_{\Sigma_c}\mathcal{L} = -\frac{1}{2}\mathbb{E}_q\left[\frac{1}{2}\left(S_{\mathbf{z}} + S_{\mathbf{z}}^T\right)\right] \tag{4.4}$$

$$\Delta\Sigma_c' = \Delta\Sigma_c + \frac{\gamma}{2}\Delta\Sigma_c\Sigma_c^{-1}\Delta\Sigma_c \tag{4.5}$$

where $S_{\mathbf{z}} = \delta(\mathbf{z})_c\pi_c\Sigma_c^{-1}\left(\mathbf{z} - \boldsymbol{\mu}_c\right)\nabla_{\mathbf{z}}^T\ell(\mathbf{z})$, and γ denotes the learning rate for Σ_c. Equation 4.4 holds as $\Delta\Sigma_c = \frac{1}{2}E_q\left[S_{\mathbf{z}}\right] = \frac{1}{2}E_q\left[S_{\mathbf{z}}^T\right]$. Motivated by Lin et al. [99], Equation 4.5 ensures the positive-definiteness of the covariance matrix, which is proved by Theorem 4.3.

Theorem 4.3. *The updated covariance matrix $\Sigma_c' = \Sigma_c + \gamma\Delta\Sigma_c'$ with the modified update rule specified in Equation 4.5 is positive-definite if Σ_c is positive-definite.*

Proof of Theorem 4.3 is provided in Section 4.5.1.

We introduce a mixing coefficient parameter ρ_c, which is updated instead of the mixing coefficient π_c, to guarantee that the updated mixing coefficients are non-negative and summed to one. π_c can be calculated using the softmax function (i.e., $\pi_c = \exp(\rho_c)/\sum_{i=1}^K \exp(\rho_i)$). We can then derive the gradient identity for the mixing coefficient parameter as follows:

Theorem 4.4. *Let ρ_c be a mixing coefficient parameter. Then, the following gradient identity holds:*

$$\nabla_{\rho_c}\mathcal{L} = \mathbb{E}_q\left[\pi_c\left(\delta(\mathbf{z})_c - 1\right)\ell(\mathbf{z})\right] \tag{4.6}$$

Proof of Theorem 4.4 is given in Section 4.5.1. Because the gradients of the latent vector with respect to the latent parameters are computed by implicit differentiation via Stein's lemma, we obtain the implicit reparameterization gradients introduced by Figurnov et al. [46].

4.2.3 Contrastive learning

We introduce new unsupervised conditional contrastive loss (U2C loss) to learn salient attributes from data and to facilitate unsupervised conditional generation. We consider a batch of latent vectors $\{\mathbf{z}^i\}_{i=1}^B \sim q(\mathbf{z})$, where B is the batch size. Generator G receives the i-th latent vector $\mathbf{z}^i$ and generates data $\mathbf{x}_g^i = G(\mathbf{z}^i)$. The adversarial loss for G with respect to the sample $\mathbf{z}^i$ is as follows:

$$\ell_{\text{adv}}(\mathbf{z}^i) = -D(G(\mathbf{z}^i)) \tag{4.7}$$

We also introduce an encoder network E to implement U2C loss. The synthesized data $\mathbf{x}_g^i$ enters E, and E generates an encoded vector $\mathbf{e}_{\mathbf{x}}^i = E(\mathbf{x}_g^i)$. Then, we find the mean vector $\boldsymbol{\mu}_C^i$, where C is the component ID with the highest responsibility $q(c|\mathbf{z}^i)$. We calculate C first because a generated sample should have the attribute of the most responsible component among multiple components in the continuous space. Second, to update the parameters of the prior using implicit reparameterization, the loss should be a function of a latent vector $\mathbf{z}^i$, as proved in Theorems 4.1, 4.2, and 4.4. The component ID for each sample is calculated as follows:

$$C^i = \underset{c}{\operatorname{argmax}}\, q(c|\mathbf{z}^i) = \underset{c}{\operatorname{argmax}}\, \delta(\mathbf{z}^i)_c \pi_c \tag{4.8}$$

where $q(c|\mathbf{z}^i) = \delta(\mathbf{z}^i)_c \pi_c$ is derived from Equation 4.1. To satisfy the assumption of the continuously differentiable loss function in Theorems 4.1 and 4.2, we adopt the Gumbel-Softmax relaxation [73], instead of the argmax function. We use $\boldsymbol{\mu}_{\mathbf{C}}^i = \sum_{c=1}^K \mathbf{C}_c^i \boldsymbol{\mu}_c$ to calculate U2C loss to ensure that the loss function is continuously differentiable with respect to $\mathbf{z}^i$, where $\mathbf{C}^i = \text{Gumbel-Softmax}_\tau(\boldsymbol{\delta}(\mathbf{z}^i)\boldsymbol{\pi})$ and $\tau = 0.01$. We derive U2C loss as follows:

$$\ell_{\text{U2C}}(\mathbf{z}^i) = -\log \frac{\exp(\cos\theta_{ii})}{\frac{1}{B}\sum_{j=1}^B \exp(\cos\theta_{ij})} \tag{4.9}$$

where we select the cosine similarity between $\mathbf{e}_{\mathbf{x}}^{i}$ and $\boldsymbol{\mu}_{\mathbf{C}}^{j}$, $\cos\theta_{ij} = \mathbf{e}_{\mathbf{x}}^{i} \cdot \boldsymbol{\mu}_{\mathbf{C}}^{j} / \|\mathbf{e}_{\mathbf{x}}^{i}\| \|\boldsymbol{\mu}_{\mathbf{C}}^{j}\|$ as the critic function that approximates the log density ratio $\log p(C^j | \mathbf{x}_g^i)/p(C^j)$ for contrastive learning.

Intuitively, a mean vector $\boldsymbol{\mu}_{\mathbf{C}}^{i}$ of a latent mixture component is regarded as a prototype of each attribute. U2C loss encourages the encoded vector $\mathbf{e}_{\mathbf{x}}^{i}$ of the generated sample to be similar to its assigned low-dimensional prototype $\boldsymbol{\mu}_{\mathbf{C}}^{i}$ in the latent space. This allows each salient attribute clusters in the latent space, and each component of the learned latent distribution is responsible for a certain attribute of the data. If $\cos\theta_{ii}$ is proportional to the log density ratio $\log p(C^i | \mathbf{x}_g^i)/p(C^i)$, minimizing U2C loss in Equation 4.9 is equivalent to maximizing the lower bound of the mutual information $I(C^i; \mathbf{x}_g^i)$, as discussed by Poole et al. [127] and Zhong et al. [177].

To help that the latent space does not learn low-level attributes, such as background color, we additionally used the SimCLR [26] loss on the generated data with DiffAugment [175] to train the encoder on colored image datasets. For the CIFAR and CelebA datasets, we used the SimCLR loss [26] for the encoder. We applied color, translation, and cutout transformations to the generated data using DiffAugment[1] [175]. The SimCLR loss is calculated using the generated data $\mathbf{x}_g^i$ and augmentation A as follows:

$$\ell_{\text{SimCLR}}(\mathbf{z}^i) = -\log \frac{\exp(E(\mathbf{x}_g^i) \cdot E(A(\mathbf{x}_g^i)) \ / \ \|E(\mathbf{x}_g^i)\| \|E(A(\mathbf{x}_g^i))\|)}{\sum_{j=1}^{B} \exp(E(\mathbf{x}_g^i) \cdot E(A(\mathbf{x}_g^j)) \ / \ \|E(\mathbf{x}_g^i)\| \|E(A(\mathbf{x}_g^j))\|)} \tag{4.10}$$

where $\mathbf{x}_g^i = G(\mathbf{z}^i)$.

The encoder E is trained to minimize $\frac{1}{B} \sum_{i=1}^{B} \left(\ell_{\text{adv}}(\mathbf{z}^i) + \ell_{\text{SimCLR}}(\mathbf{z}^i) \ + \lambda \ell_{\text{U2C}}(\mathbf{z}^i) \right)$, where λ denotes the coefficient of U2C loss. The generator G is trained to minimize $\frac{1}{B} \sum_{i=1}^{B} \left(\ell_{\text{adv}}(\mathbf{z}^i) + \lambda \ell_{\text{U2C}}(\mathbf{z}^i) \right)$. Both $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are learned by substituting $\ell_{\text{adv}}(\mathbf{z}^i) + \lambda \ell_{\text{U2C}}(\mathbf{z}^i)$ into ℓ of Equations 4.2 and 4.5, respectively. When U2C loss is

[1] https://github.com/mit-han-lab/data-efficient-gans

used to update $\boldsymbol{\pi}$, U2C loss hinders $\boldsymbol{\pi}$ from estimating the imbalance ratio of attributes in the data well, which is discussed in Section 4.4.2 with a detailed explanation and an empirical result. Therefore, $\boldsymbol{\rho}$, from which $\boldsymbol{\pi}$ is calculated, uses only the adversarial loss, and ℓ of Equation 4.6 is substituted by $\ell_{\text{adv}}(\mathbf{z}^i)$. $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$ and $\boldsymbol{\rho}$ are learned using a batch average of estimated gradients, which is referred to as stochastic gradient estimation, instead of expectation over the latent distribution q. In Section 4.5.1, we discuss the assumptions required to use the gradient identities in Theorems 4.1 and 4.2. The entire training procedure of SLOGAN is presented in Algorithm 4.1.

After training SLOGAN, unconditional generation can be performed by sampling $\mathbf{z}$ from the Gaussian mixture $q(\mathbf{z})$, then generating a sample $G(\mathbf{z})$. SLOGAN can also perform unsupervised conditional generation by selecting a component c, sampling $\mathbf{z}$ from the mixture component $q(\mathbf{z}|c)$, and generating a sample $G(\mathbf{z})$.

4.2.4 Cluster assignment

Given a test data, the probability for each cluster can be calculated using the assumption of the critic function, which enables us to assign a cluster for the data. In Section 4.2.3, we chose $\cos\theta_{ic}$ as the critic function assuming that it is proportional to $\log p(c|\mathbf{x}_g)$. If the real data distribution $p(\mathbf{x}_r)$ and the generator distribution $p(\mathbf{x}_g)$ are sufficiently similar via adversarial learning, the cosine similarity between $E(\mathbf{x}_r)$ and $\boldsymbol{\mu}_c$ can also be considered proportional to $\log p(c|\mathbf{x}_r)$. Therefore, for real data, we obtain the probability for each cluster as follows:

$$\hat{p}(c|\mathbf{x}_r) = \frac{\exp(\cos\theta_c)}{\sum_{k=1}^{K}\exp(\cos\theta_k)} \tag{4.11}$$

where $\cos\theta_k = E(\mathbf{x}_r)\cdot\boldsymbol{\mu}_k/\|E(\mathbf{x}_r)\|\|\boldsymbol{\mu}_k\|$ is the cosine similarity between $E(\mathbf{x}_r)$ and $\boldsymbol{\mu}_k$. The data can then be assigned to the cluster with the highest probability (i.e., $\text{argmax}_c\,\hat{p}(c|\mathbf{x}_r)$).

Algorithm 4.1 Training procedure of SLOGAN

```
Initialize $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$, $\boldsymbol{\rho}$, parameters of $D$, $G$, and $E$
while training loss is not converged do
    Sample a batch of data $\{\mathbf{x}^i\}_{i=1}^B \sim p(\mathbf{x})$
    Sample a batch of latent vectors $\{\mathbf{z}^i\}_{i=1}^B \sim q(\mathbf{z})$
    for $i = 1, ..., B$ do
        Calculate $\ell_{\text{adv}}(\mathbf{z}^i)$ and $\ell_{\text{U2C}}(\mathbf{z}^i)$ for a latent vector $\mathbf{z}^i$
        $S_{\mathbf{z}^i} \leftarrow \delta(\mathbf{z}^i)_c \pi_c \Sigma_c^{-1} (\mathbf{z}^i - \boldsymbol{\mu}_c) \nabla_{\mathbf{z}^i}^T (\ell_{\text{adv}}(\mathbf{z}^i) + \lambda \ell_{\text{U2C}}(\mathbf{z}^i))$
    end for
    for $c = 1, ..., K$ do
        Update $\boldsymbol{\mu}_c$, $\boldsymbol{\Sigma}_c$ and $\boldsymbol{\rho}_c$ via stochastic gradient estimation
          $\boldsymbol{\mu}_c \leftarrow \boldsymbol{\mu}_c - \gamma \frac{1}{B} \sum_{i=1}^B \delta(\mathbf{z}^i)_c \pi_c \nabla_{\mathbf{z}^i} (\ell_{\text{adv}}(\mathbf{z}^i) + \lambda \ell_{\text{U2C}}(\mathbf{z}^i))$
          $\Delta\Sigma_c \leftarrow -\frac{1}{4B} \sum_{i=1}^B (S_{\mathbf{z}^i} + S_{\mathbf{z}^i}^T)$
          $\Sigma_c \leftarrow \Sigma_c + \gamma (\Delta\Sigma_c + \frac{\gamma}{2} \Delta\Sigma_c \Sigma_c^{-1} \Delta\Sigma_c)$
          $\rho_c \leftarrow \rho_c - \gamma \frac{1}{B} \sum_{i=1}^B \pi_c (\delta(\mathbf{z}^i)_c - 1) \ell_{\text{adv}}(\mathbf{z}^i)$
    end for
    Update $G$, $E$ and $D$ using SGD
      $\nabla_{G,E} \frac{1}{B} \sum_{i=1}^B (\ell_{\text{adv}}(\mathbf{z}^i) + \lambda \ell_{\text{U2C}}(\mathbf{z}^i))$
      $\nabla_D \left( -\frac{1}{B} \sum_{i=1}^B \ell_{\text{adv}}(\mathbf{z}^i) - \frac{1}{B} \sum_{i=1}^B D(\mathbf{x}^i) \right)$
end while
```

Algorithm 4.2 Attribute manipulation

```
Initialize probe data with the desired attribute $\mathbf{X}_{c=1} \leftarrow \{\mathbf{x}_{c=1}^i\}_{i=1}^M$ and $\bar{\mathbf{X}}_{c=1} \leftarrow \{\mathbf{x}_{c=1}^i\}_{i=1}^M$
Initialize probe data without the desired attribute $\mathbf{X}_{c=0} \leftarrow \{\mathbf{x}_{c=0}^i\}_{i=1}^M$ and $\bar{\mathbf{X}}_{c=0} \leftarrow \{\mathbf{x}_{c=0}^i\}_{i=1}^M$
for each mixup iteration $t$ in $\{1, ..., T\}$ do
    $\bar{\mathbf{X}}_{c=1} \leftarrow \bar{\mathbf{X}}_{c=1} \cup \textsc{Mixup}(\mathbf{X}_{c=1}, \textsc{Permute}(\mathbf{X}_{c=1}))$
    $\bar{\mathbf{X}}_{c=0} \leftarrow \bar{\mathbf{X}}_{c=0} \cup \textsc{Mixup}(\mathbf{X}_{c=0}, \textsc{Permute}(\mathbf{X}_{c=0}))$
end for
for each augmented data index $j$ in $\{1, ..., M(T+1)\}$ do
    $\cos\theta_{00}^j \leftarrow E(\bar{\mathbf{x}}_{c=0}^j) \cdot \boldsymbol{\mu}_0 / \|E(\bar{\mathbf{x}}_{c=0}^j)\| \|\boldsymbol{\mu}_0\|$
    $\cos\theta_{01}^j \leftarrow E(\bar{\mathbf{x}}_{c=0}^j) \cdot \boldsymbol{\mu}_1 / \|E(\bar{\mathbf{x}}_{c=0}^j)\| \|\boldsymbol{\mu}_1\|$
    $\cos\theta_{10}^j \leftarrow E(\bar{\mathbf{x}}_{c=1}^j) \cdot \boldsymbol{\mu}_0 / \|E(\bar{\mathbf{x}}_{c=1}^j)\| \|\boldsymbol{\mu}_0\|$
    $\cos\theta_{11}^j \leftarrow E(\bar{\mathbf{x}}_{c=1}^j) \cdot \boldsymbol{\mu}_1 / \|E(\bar{\mathbf{x}}_{c=1}^j)\| \|\boldsymbol{\mu}_1\|$
end for
$\mathcal{L}_{\text{m}} = -\frac{1}{M(T+1)} \sum_{j=1}^{M(T+1)} \log \frac{\exp(\cos\theta_{00}^j) + \exp(\cos\theta_{11}^j)}{\exp(\cos\theta_{00}^j) + \exp(\cos\theta_{01}^j) + \exp(\cos\theta_{10}^j) + \exp(\cos\theta_{11}^j)}$
Minimize $\mathcal{L}^p$ with respect to $E$, $G$, and $\boldsymbol{\mu}$
```

4.2.5 Attribute manipulation

For datasets such as face attributes, a data point can have multiple attributes simultaneously. To learn a desired attribute from such data, a probe dataset $\{\mathbf{x}_c^i\}_{i=1}^M$ for the c-th latent component, which consists of M data points with the desired attribute, can be utilized. We propose the following loss:

$$\mathcal{L}_{\mathrm{m}} = \frac{1}{M} \sum_{i=1}^{M} -\log \frac{\exp(\cos \theta_c^i)}{\sum_{k=1}^{K} \exp(\cos \theta_k^i)} \tag{4.12}$$

where $\cos \theta_k^i = E(\mathbf{x}_c^i) \cdot \boldsymbol{\mu}_k / \|E(\mathbf{x}_c^i)\| \|\boldsymbol{\mu}_k\|$ is the cosine similarity between $E(\mathbf{x}_c^i)$ and $\boldsymbol{\mu}_k$. Our model manipulates attributes by minimizing $\mathcal{L}_{\mathrm{m}}$ for $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$, G, and E. In addition, we utilized mixup [173] to make the best use of a small amount of probe data when manipulating attributes. Algorithm 4.2 describes the procedure for using mixup for attribute manipulation when $K = 2$. The number of iterations for the mixup (T) was set to five. The advantage of SLOGAN in attribute manipulation is that it can learn imbalanced attributes even if the attributes in the probe dataset are balanced, and perform better conditional generation.

4.3 Experimental Setup

4.3.1 Datasets

We used the MNIST [93], Fashion-MNIST (FMNIST) [169], CIFAR-10 [88], CelebA [101], CelebA-HQ [78], and AFHQ [29] datasets to evaluate the proposed method. We also constructed some datasets with imbalanced attributes. For example, we used two classes of the MNIST dataset (0 vs. 4, referred to as MNIST-2), two classes of the CIFAR-10 dataset (frogs vs. planes, referred to as CIFAR-2), and five clusters of the FMNIST dataset ({Trouser}, {Bag}, {T-shirt/top, Dress}, {Pullover, Coat, Shirt}, {Sneaker, Sandal, Ankle Boot}, referred to as FMNIST-5 with an imbalance ratio of 1:1:2:3:3). In addition to various datasets, we also used the 10x_73k dataset [176], which consists of RNA transcript counts. From the results of the clustering performances on the 10x 73k dataset, we show that SLOGAN learns useful imbalanced attributes and can be helpful in the use of unlabeled biomedical data. Details of the datasets are provided in Section 4.5.2.

Although SLOGAN and other methods do not utilize labels for training, the data in experimental settings have labels predefined by humans. We consider that each class of dataset contains a distinct attribute. Thus, the model performance was measured using classes of datasets. The number of latent components or the dimension of the discrete latent code (K) was set as the number of classes of data.

4.3.2 Evaluation metrics

The performance of our method was evaluated quantitatively in three aspects: (1) whether the model could learn distinct attributes and cluster real data (i.e., cluster assignment), which is evaluated using normalized mutual information (NMI) [116], (2) whether the overall data distribution $p(\mathbf{x}_r)$ could be estimated (i.e., unconditional data generation), which is measured using the Fréchet inception distance (FID) [66], and, most importantly, (3) whether the data distribution for each attribute $p(\mathbf{x}_r|c)$ could be estimated (i.e., unsupervised conditional generation).

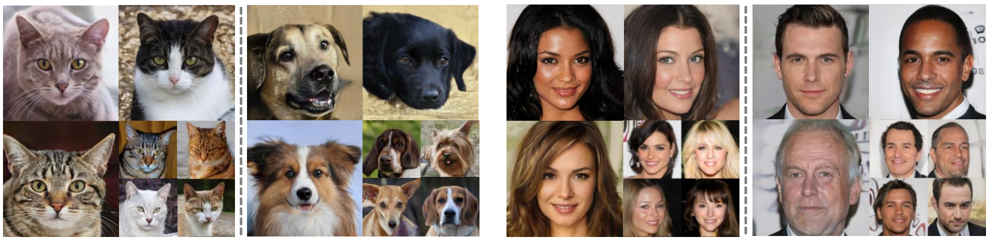
For unsupervised conditional generation, it is important to account for intra-cluster diversity as well as the quality of the generated samples. We introduce a modified version of FID named intra-cluster Fréchet inception distance (ICFID) described in Algorithm 4.3. We calculate FIDs between the real data of each class and generated data from each latent code (a mixture component for DeLiGAN and SLOGAN, and a category for other methods). We then greedily match a latent code with a class of real data with the smallest FID. We define ICFID as the average FID between the matched pairs and use it as an evaluation metric for unsupervised conditional generation. ICFID additionally provides class-cluster assignment (i.e., which cluster is the closest to the class).

Algorithm 4.3 Intra-cluster FID

```
input  : $\{\{\mathbf{x}_y^i\}_{i=1}^N\}_{y=1}^K$ - Data sampled from $p(\mathbf{x}|y)$ for $y = 1, ..., K$;
         $\{\{\mathbf{z}_c^i\}_{i=1}^N\}_{c=1}^K$ - Latent vectors sampled from $q(\mathbf{z}|c)$ for $c = 1, ..., K$
output : ICFID - Intra-cluster FID;
         $Y_c$ - Class-cluster assignments
  $Y \leftarrow \{1, ..., K\}$
  $C \leftarrow \{1, ..., K\}$
  for each class $y$ in $Y$ do
      $\mathbf{X}_r \leftarrow \{\mathbf{x}_y^i\}_{i=1}^N$
      for each cluster $c$ in $C$ do
          $\mathbf{X}_g \leftarrow \{\mathbf{x}_c^i\}_{i=1}^N$
          $d(y, c) \leftarrow \text{FID}(\mathbf{X}_r, \mathbf{X}_g)$
      end for
      $c^* \leftarrow \text{argmin}_{c \in C}\, d(y, c)$
      $\text{ICFID}(y) \leftarrow d(y, c^*)$
      $Y_c(y) \leftarrow c^*$
      Remove $c^*$ from $C$
  end for
  $\text{ICFID} \leftarrow \frac{1}{K} \sum_{y=1}^{K} \text{ICFID}(y)$
```

Table 4.1: Performance comparison on balanced attributes

Dataset	Metric	InfoGAN	DeLiGAN	DeLiGAN+	ClusterGAN	SCGAN	CD-GAN	PGMGAN	**SLOGAN**
MNIST	NMI	0.90±0.03	0.70±0.05	0.77±0.05	0.81±0.02	0.74±0.06	0.87±0.03	0.16±0.27	**0.92±0.00**
	FID	1.72±0.17	1.92±0.12	2.00±0.16	1.71±0.07	3.06±0.53	2.75±0.04	5.76±1.67	**1.67±0.15**
	ICFID	5.56±0.71	5.74±0.25	5.64±0.39	5.12±0.07	16.65±2.01	7.03±0.23	53.40±12.49	**4.99±0.19**
FMNIST	NMI	0.64±0.02	0.64±0.03	0.57±0.07	0.61±0.03	0.56±0.01	0.56±0.04	0.47±0.01	**0.66±0.01**
	FID	5.28±0.12	6.65±0.48	7.23±0.56	6.32±0.25	**5.07±0.19**	9.05±0.11	9.13±0.28	5.20±0.36
	ICFID	32.18±2.11	34.87±5.29	30.53±8.71	37.20±5.50	26.23±7.10	36.61±0.47	40.00±4.38	**23.31±2.77**
CIFAR-2	NMI	0.05±0.03	0.15±0.13	0.12±0.12	0.34±0.02	0.00±0.00	0.38±0.01	0.67±0.00	**0.78±0.03**
	FID	58.84±13.11	338.97±70.85	116.95±19.42	36.28±1.12	39.44±1.72	34.45±0.74	29.49±0.51	**28.99±0.36**
	ICFID	91.97±14.21	361.66±71.28	153.19±17.71	47.02±1.85	71.54±5.41	43.98±1.47	**35.67±0.61**	35.68±0.51
CIFAR-10	NMI	0.03±0.00	0.06±0.00	0.09±0.04	0.10±0.00	0.01±0.00	0.03±0.01	0.29±0.02	**0.34±0.01**
	FID	81.84±2.27	212.20±4.52	110.51±7.70	61.97±3.69	199.28±57.16	34.13±1.13	31.50±0.73	**20.61±0.40**
	ICFID	139.20±2.09	305.32±5.05	215.63±11.16	124.27±5.95	262.54±59.29	95.43±3.58	81.25±11.55	**71.23±6.76**



(a) AFHQ (1:2)

(b) CelebA-HQ (1.7:1)

Figure 4.3: Generated high-fidelity images from SLOGAN on (a) AFHQ and (b) CelebA-HQ.

Table 4.2: Performance comparison on imbalanced attributes

Dataset	Metric	InfoGAN	DeLiGAN	DeLiGAN+	ClusterGAN	SCGAN	CD-GAN	PGMGAN	**SLOGAN**
MNIST-2 (7:3)	NMI	0.28±0.19	0.90±0.04	0.48±0.09	0.27±0.19	0.67±0.11	0.41±0.03	0.79±0.21	**0.92±0.05**
	FID	4.92±0.85	4.21±0.84	4.63±2.02	4.25±1.06	4.34±0.73	4.67±1.92	8.90±14.82	**4.02±0.86**
	ICFID	36.35±10.65	25.34±1.72	26.61±1.49	25.41±1.02	16.47±1.51	26.71±2.47	14.82±9.16	**5.91±1.06**
FMNIST-5	NMI	0.58±0.07	**0.68±0.05**	0.65±0.01	0.60±0.02	0.60±0.06	0.59±0.01	0.24±0.02	0.66±0.06
	FID	5.40±0.14	7.05±0.49	6.33±0.44	5.61±0.17	**5.01±0.20**	9.34±0.56	11.80±0.43	5.29±0.16
	ICFID	43.69±10.84	36.21±3.07	35.41±0.79	36.94±5.81	44.48±21.62	39.31±1.18	77.30±8.60	**32.46±3.18**
10x_73k	NMI	0.42±0.06	0.61±0.01	0.60±0.01	0.66±0.02	0.47±0.02	0.68±0.03	0.33±0.07	**0.76±0.02**
CIFAR-2 (7:3)	NMI	0.05±0.01	0.00±0.00	0.03±0.03	0.22±0.02	0.00±0.00	0.22±0.03	0.42±0.03	**0.69±0.02**
	FID	51.30±2.53	131.73±50.98	115.19±17.95	36.62±2.16	45.28±1.81	36.40±1.01	29.76±1.65	**29.09±0.73**
	ICFID	88.49±6.85	186.31±28.31	173.81±18.29	75.52±4.82	88.58±4.57	76.91±1.07	57.06±3.31	**45.83±3.03**
CIFAR-2 (9:1)	NMI	0.00±0.00	0.02±0.02	0.09±0.11	0.02±0.01	0.00±0.00	0.05±0.03	0.16±0.03	**0.38±0.01**
	FID	60.76±8.97	129.50±25.33	139.75±47.13	41.69±0.83	50.45±1.56	38.15±2.70	30.23±1.31	**29.47±1.53**
	ICFID	138.24±10.23	205.26±10.93	196.00±17.86	133.31±2.03	123.35±6.56	128.46±3.03	101.68±3.87	**86.75±1.87**

4.4 Experimental Results

4.4.1 Evaluation results

We compared SLOGAN with InfoGAN [27], DeLiGAN [59], ClusterGAN [116], Self-conditioned GAN (SCGAN) [100], CD-GAN [120], and PGMGAN [9]. DeLiGAN has no encoder network; hence the pre-activation of the penultimate layer of D was used for the clustering metrics. For a fair comparison, we also compared DeLiGAN with an encoder network (referred to as DeLiGAN+). The same network architecture and hyperparameters (e.g., learning rate) were used across all methods for comparison. Details of the experiments and DeLiGAN+ are presented in Section 4.5.2.

Balanced attributes We compare SLOGAN with existing unsupervised conditional GANs on datasets with balanced attributes. As shown in Table 4.1, SLOGAN outperformed other GANs, and comparisons with methods with categorical priors (ClusterGAN and CD-GAN) verified the advantages of the mixture priors.

Imbalanced attributes In Table 4.2, we compare SLOGAN with existing methods on datasets with imbalanced attributes. ICFIDs of our method are much better than those of other methods, which indicates that SLOGAN was able to robustly capture the minority attributes in datasets and can generate data conditioned on the learned

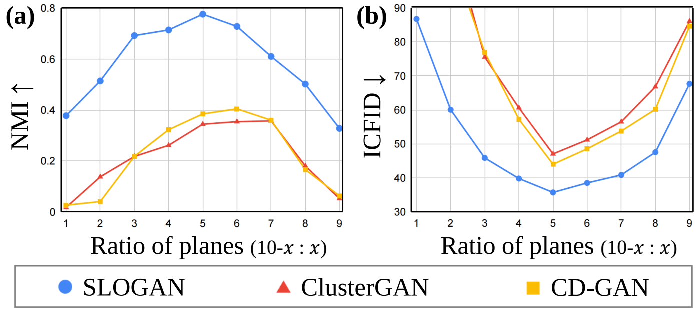


Figure 4.4: Performance comparison with respect to the imbalance ratio on (a) cluster assignment and (b) unsupervised conditional generation.

attributes. In CIFAR-2 (7:3), the ratio of frog and plane is 7 to 3 and the estimated $\boldsymbol{\pi}$ is (0.69±0.02, 0.31±0.02), which are very close to the ground-truth (0.7, 0.3). Figure 4.3 (a) shows the images generated from each latent component of SLOGAN on AFHQ (Cat:Dog=1:2). More qualitative results are presented in Section 4.4.4.

Performance with respect to imbalance ratio We compared the performance of SLOGAN with competitive benchmarks (ClusterGAN and CD-GAN) by changing the imbalance ratios of CIFAR-2 from 9:1 to 1:9. SLOGAN showed higher performance than the benchmarks on cluster assignment (Figure 4.4 (a)) and unsupervised conditional generation (Figure 4.4 (b)) for all imbalance ratios. Furthermore, our method shows a larger gap in ICFID with the benchmarks when the ratio of planes is low. This implies that SLOGAN works robustly in situations in which the attributes of data are highly imbalanced.

Benefits of ICFID DeLiGAN and ClusterGAN trained on the MNIST-2 (7:3) exhibited comparable FIDs to SLOGAN (DeLiGAN: 4.21, ClusterGAN: 4.25, and SLOGAN: 4.02); however they showed ICFIDs approximately four times higher (DeLiGAN: 25.34, ClusterGAN: 25.61, and SLOGAN: 5.91). From the data generated from each latent component of DeLiGAN and ClusterGAN in Figure 4.5 (b) and (c), we confirm that the attributes were not learned well in the latent space of DeLiGAN. By contrast, from the data generated from each latent component of SLOGAN presented in Figure 4.5 (a), SLOGAN successfully learned the attributes in its latent space. This shows that ICFID is useful for evaluating the performance of unsupervised conditional generation. In addition, ICFID can evaluate the diversity of images generated from a discrete latent code or mode because ICFID is based on FID. As shown in Figure 4.6, when a mode collapse occurs, the diversity of samples decreases drastically, and DeLiGAN trained on the CIFAR-2 (7:3) shows approximately twice the ICFID than those of InfoGAN and ClusterGAN (DeLiGAN: 186.31, InfoGAN: 88.49, and ClusterGAN: 75:52).

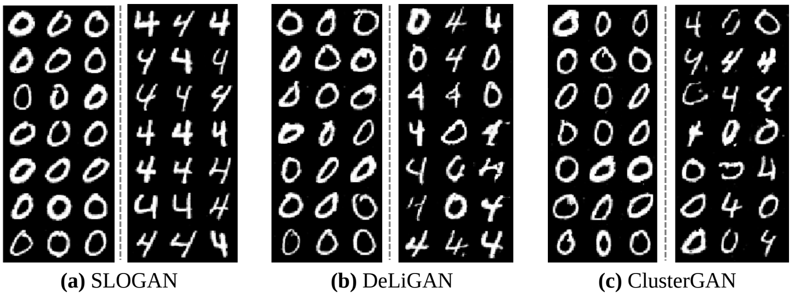

(a) SLOGAN **(b)** DeLiGAN **(c)** ClusterGAN

Figure 4.5: An example where ICFID is useful. The left and right images show generated images from each latent code of (a) SLOGAN, (b) DeLiGAN and (c) ClusterGAN trained on the MNIST-2 (7:3) dataset, respectively.

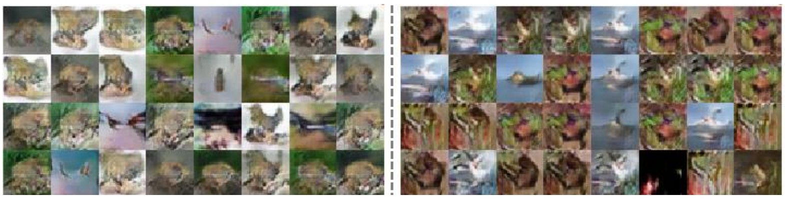

DeLiGAN

Figure 4.6: Another example where ICFID is useful. The left and right images show generated images from each latent component of DeLiGAN trained on the CIFAR-2 (7:3) dataset.

Table 4.3: Effectiveness of U2C loss

Dataset	Ablation	NMI ↑	FID ↓	ICFID ↓
Synthetic	SLOGAN w/o ℓ_{U2C}	-	-	✗
	SLOGAN	-	-	✓
MNIST-2 (7:3)	SLOGAN w/o ℓ_{U2C}	0.25	4.62	9.43
	SLOGAN	**0.92**	**4.02**	**5.91**
FMNIST-5	SLOGAN w/o ℓ_{U2C}	0.14	**5.27**	43.15
	SLOGAN	**0.66**	5.29	**32.46**
CIFAR-2	SLOGAN w/o ℓ_{U2C}	0.01	29.18	41.72
	SLOGAN	**0.78**	**28.99**	**35.68**
CIFAR-2 (7:3)	SLOGAN w/o ℓ_{U2C}	0.08	30.34	48.82
	SLOGAN	**0.69**	**29.09**	**45.83**
CIFAR-10	SLOGAN w/o ℓ_{U2C}	0.08	20.91	78.26
	SLOGAN	**0.34**	**20.61**	**71.23**

4.4.2 Ablation study

U2C loss Table 4.3 shows the benefit of U2C loss on several datasets. Low-level features (e.g., color) of the CIFAR dataset differ depending on the class, which enables SLOGAN to function to some extent without U2C loss on CIFAR-10. In the MNIST dataset, the colors of the background (black) and object (white) are the same, and only the shape of objects differs depending on the class. U2C loss played an essential role on MNIST (7:3). The modes of the Synthetic dataset (Figure 4.1) are placed adjacent to each other, and SLOGAN cannot function on this dataset without U2C loss. From the results, we observed that the effectiveness of U2C loss depends on the properties of the datasets.

Factor analysis Table 4.4 shows the ablation study on SLOGAN trained with CIFAR-2 (7:3). $\pi_{y=0}$ and $\pi_{y=1}$ represent the mixing coefficients of the latent components that correspond to the frogs and planes, respectively, and the ground-truth of $\pi_{y=0}$ is 0.7. Rows 1-6 of Table 4.4 compare the performance depending on the factors affecting the performance of SLOGAN ($\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$, $\boldsymbol{\rho}$ updates, and ℓ_{U2C}). We confirmed that SLOGAN

Table 4.4: Ablation study on CIFAR-2 (7:3)

Ablation	$\pi_{y=0}$ (ground-truth: 0.7)	ICFID ↓
Factor analysis		
SLOGAN without $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$, $\boldsymbol{\rho}$ updates, ℓ_{U2C}	0.50	84.44
SLOGAN without $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$, $\boldsymbol{\rho}$ updates	0.50	77.32
SLOGAN without $\boldsymbol{\mu}$ update	0.52	73.79
SLOGAN without $\boldsymbol{\rho}$ update	0.50	63.09
SLOGAN without $\boldsymbol{\Sigma}$ update	0.69	48.34
SLOGAN without ℓ_{U2C}	0.66	48.82
Implicit reparameterization		
DeLiGAN with ℓ_{U2C}	0.50	60.51
DeLiGAN with ℓ_{U2C} and implicit $\boldsymbol{\rho}$ update	1.00	86.48
Loss for $\boldsymbol{\rho}$ update		
SLOGAN with ℓ_{U2C} for $\boldsymbol{\rho}$ update	0.62	52.67
SimCLR analysis		
SLOGAN without SimCLR	0.66	49.25
SLOGAN without SimCLR on real data only	0.67	48.41
SLOGAN without SimCLR on both real and fake data	0.69	47.93
Attribute manipulation		
SLOGAN with probe data	0.71	44.97
SLOGAN with probe data and mixup	0.70	44.26
SLOGAN	0.69	45.83

with all the factors demonstrated the highest performance. Among the parameters of the latent distribution, the $\boldsymbol{\mu}$ update leads to the highest performance improvement. ICFID of SLOGAN without $\boldsymbol{\rho}$ update (the 4th row of Table 4.4) indicates that SLOGAN outperformed existing unsupervised conditional GANs even when assuming a uniform distribution of the attributes.

Implicit reparameterization To show the advantage of implicit over explicit reparameterization, we implemented DeLiGAN with U2C loss by applying explicit reparameterization on $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$. Because the mixing coefficient cannot be updated with explicit reparameterization to the best of our knowledge, we also implemented DeLiGAN with U2C loss and implicit reparameterization on $\boldsymbol{\rho}$ using Equation 4.6. In rows 7-8 of Table 4.4, SLOGAN using implicit reparameterization outperformed explicit reparameterization. When implicit $\boldsymbol{\rho}$ update was added, the prior collapsed into a single component ($\pi_{y=0} = 1$) and ICFID increased. The lower variance of implicit reparameterized gradients prevents the prior from collapsing into a single component and improves the performance.

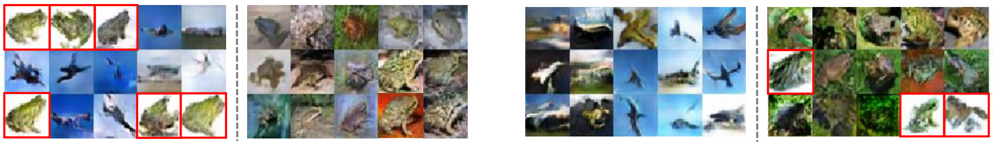
Loss for $\boldsymbol{\rho}$ update We do not use U2C loss ℓ_{U2C} to learn the mixing coefficient parameters $\boldsymbol{\rho}$. We construct U2C loss to approximate the negative mutual information $-I(C; \mathbf{x}_g)$ that can be decomposed into entropy and conditional entropy as follows:

$$\ell_{\mathrm{U2C}}(\mathbf{z}) \approx -I(C; \mathbf{x}_g) = H(C|\mathbf{x}_g) - H(C) \tag{4.13}$$

The conditional entropy term reduces the uncertainty of the component from which the generated data are obtained. The entropy term promotes that component IDs are uniformly distributed. In terms of $\boldsymbol{\rho}$ update, the entropy term $H(C)$ drives $p(C)$ toward a discrete uniform distribution. Therefore, using ℓ_{U2C} for learning $\boldsymbol{\rho}$ pulls $\boldsymbol{\pi}$ to a discrete uniform distribution and can hinder the learned $\boldsymbol{\pi}$ from accurately estimating the imbalance ratio inherent in the data. In the 9th row of Table 4.4, we observed that the unsupervised conditional generation performance was undermined and the estimated imbalance ratio ($\pi_{y=0}$) was learned closer to a discrete uniform distribution (0.5) when ℓ_{U2C} was used for $\boldsymbol{\rho}$ update.

SimCLR analysis For the colored image datasets, SLOGAN uses the SimCLR loss for the encoder with only fake (generated) data to further enhance the unsupervised conditional generation performance. The 10th to 12th rows of Table 4.4 show several ablation studies that analyzed the effect of SimCLR loss on SLOGAN. SLOGAN without SimCLR still showed at least approximately 35% performance improvement compared to the existing unsupervised conditional GANs (ICFID of ClusterGAN: 75.52, CD-GAN: 76.91 in Table 4.2), even considering the fair computational cost and memory consumption. The SimCLR loss shows the highest performance improvement especially when applied only to fake data. SimCLR improved the performance by 7%.

Attribute manipulation As shown in the 13th and 14th rows of Table 4.4, probe data significantly improved the performance of SLOGAN on CIFAR-2 (7:3) with 10 probe data for each latent component. We also confirmed that the mixup applied to the probe data further enhanced the overall performance of our model. Figure 4.7 (a) shows the data generated from SLOGAN trained on CIFAR-2 (9:1) without the probe data. With extremely imbalanced attributes, SLOGAN mapped frog images with a white background onto the same component as airplanes in its latent space. When we use 10 probe data for each latent component, as shown in Figure 4.7 (b), frogs with a white background were generated from the same latent component as the other frog images.


(a) SLOGAN (without probe data)

(b) SLOGAN with probe data & mixup

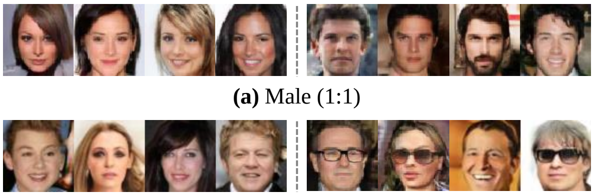
Figure 4.7: Effects of attribute manipulation on unsupervised conditional generation. Left and right images visualize generated images from different latent components. The red boxes indicate generated frog images with a white background.

Feature scale We introduced the feature scale s described in Section 4.5.2 to reinforce the discriminative power of U2C loss. For the MNIST-2 (7:3) dataset, s was set to 4. Such a parameter configuration is justified by a greedy search in $[0.5, 1, 2, 4, 8]$. The performances of SLOGAN on MNIST-2 (7:3) with different feature scales are shown in Table 4.5.

Table 4.5: Ablation study on feature scale

s	0.5	1	2	4	8
ICFID ↓	14.18	17.03	6.65	**5.91**	33.98

Intuitively, increasing the feature scale s makes the samples generated from the



(a) Male (1:1)

(b) Eyeglasses (14:1)

Figure 4.8: Qualitative results of SLOGAN on CelebA.

Table 4.6: Quantitative results of SLOGAN on CelebA

	Male	Eyeglasses
Imb. ratio	(1:1)	(14:1)
NMI ↑	0.65±0.01	0.29±0.07
FID ↓	5.18±0.20	5.83±0.44
ICFID ↓	11.00±0.66	35.57±5.10
$\pi_{y=0}$	0.56±0.02	0.82±0.04

same component closer to each other in the embedding space. From these results, we observed that the optimal choice of the temperature factor enhances the discriminative power of U2C loss.

4.4.3 Effects of probe data

CelebA + ResGAN We demonstrate that SLOGAN can learn the desired attributes using a small amount of probe data. Among multiple attributes which co-exist in the CelebA dataset, we chose Male (1:1) and Eyeglasses (14:1). We randomly selected 30 probe images for each latent component. $\pi_{y=0}$ represents the learned mixing coefficient that correspond to the latent component associated with faces without the attribute. As shown in Figure 4.8 and Table 4.6, we observed that SLOGAN learned the desired attributes.

CelebA-HQ + StyleGAN2 StyleGAN2 [80] differs from other GANs in that the latent vectors are used for *style*. Despite this difference, the implicit reparameterization and U2C loss can be applied to the input space of the mapping network. On the CelebA-HQ dataset, we used 30 male and 30 female faces as probe data. As shown in Figure 4.3 (b), SLOGAN successfully performed on high-resolution images and a recent architecture, even simultaneously with imbalanced attributes.

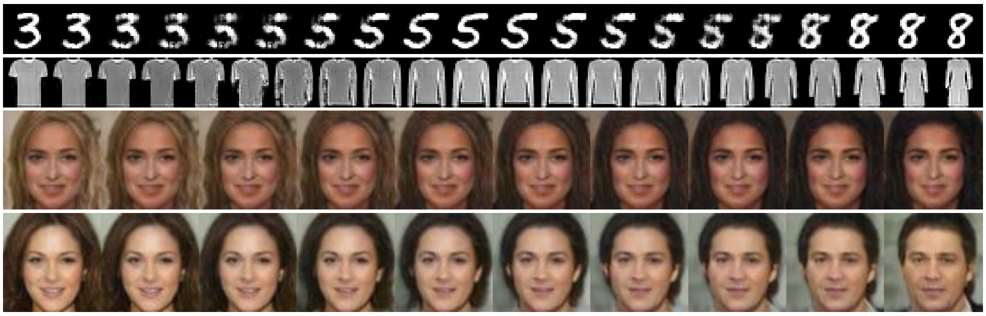


Figure 4.9: Interpolation in the latent space of the proposed method. For the MNIST and Fashion-MNIST datasets, we selected three mean vectors in the latent space and generated images from linearly interpolated latent vectors. For the CelebA dataset, we used 30 probe data and mixup for each latent component with attributes such as Black hair (3:1) and Male (1:1).

4.4.4 Qualitative results

Interpolation in latent space We also qualitatively show that the continuous nature of the prior distribution of SLOGAN makes superbly smooth interpolation possible in the latent space. In Figure 4.9, we visualize images generated from latent vectors obtained via interpolation among the mean vectors of the trained latent components. The generated images gradually changed to 3, 5, and 8 for the MNIST dataset, and t-shirt/top, pullover, and dress for the Fashion-MNIST dataset. In particular, we confirmed that the face images generated from the model trained with the CelebA data changed smoothly. The continuous attributes of real-world data are well mapped to the continuous latent space assumed by us, unlike most other methods using separated latent spaces induced via discrete latent codes.

Generated images and latent spaces Figures 4.10, 4.11, 4.15, 4.16, 4.17, the left plot of Figure 4.13, and the upper plot of Figure 4.14 show the images generated from each latent component of SLOGAN trained on various datasets. Figure 4.12, the right plot of Figure 4.13, and the lower plot of Figure 4.14 visualize 1,000 latent vectors of SLOGAN trained on the MNIST, Fashion-MNIST, MNIST-2 (7:3), and FMNIST-

5 datasets using 3D principal component analysis (PCA). Each color represents the component with the highest responsibility, and each image shows the generated image from the latent vector. As shown in Figure 4.12, similar attributes (e.g., 4, 7, and 9) are mapped to nearby components in the latent space.

Comparisons with the most recent methods We compare our method with the most recent methods such as CD-GAN [120] and PGMGAN [9] on the CIFAR-2 (7:3) dataset. From the results shown in Figure 4.18, we qualitatively confirm that SLOGAN learns imbalanced attributes of the dataset most robustly.

Highly imbalanced multi-class data We trained our method on highly imbalanced multi-class datasets by setting class 8 of the MNIST dataset to very low proportions of the other nine classes (e.g., 10:10:10:10:10:10:10:10:1:10 and 10:10:10:10:10:10:10:10:2:10). When class 8 is 0.1 fraction of the other nine classes, images of class 7 with a horizontal line outnumber images of class 8, and SLOGAN identifies 7 with a horizontal line as a more salient attribute than 8 as shown in the red boxes in Figure 4.19 (a). On the other hand, when class 8 is 0.2 fraction of the other nine classes, images of class 8 outnumber images of class 7 with a horizontal line. Therefore, SLOGAN successfully identifies 8 as a salient attribute as shown in the red box in Figure 4.19 (b).

Qualitative analysis with various imbalance ratios Figure 4.20 shows generated images from each latent component of SLOGAN trained on the AFHQ dataset. For various imbalance ratios of cats and dogs, we qualitatively analyze SLOGAN without using probe data. When the imbalance ratios are 1:1 and 1:2, SLOGAN identifies cat/dog as the most salient attribute and learned the attribute successfully as presented in Figure 4.20 (a) and (b). When the imbalance ratio is 1:5, SLOGAN discovers folded ears as the most salient attribute as shown in Figure 4.20 (c).

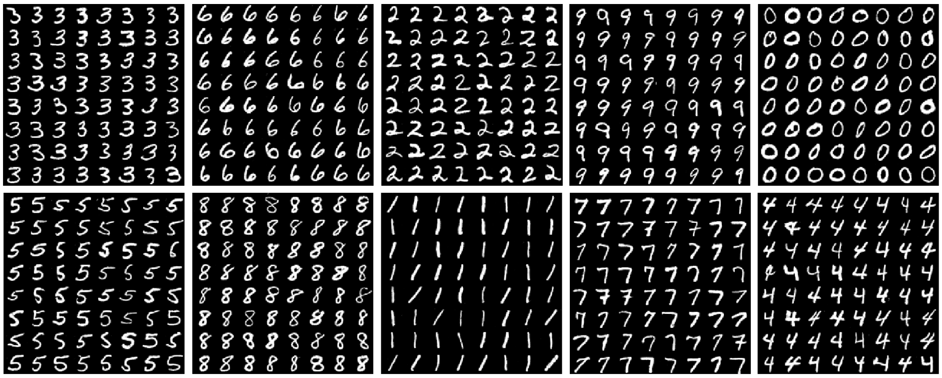


Figure 4.10: Generated images from each latent component of SLOGAN trained on the MNIST dataset.

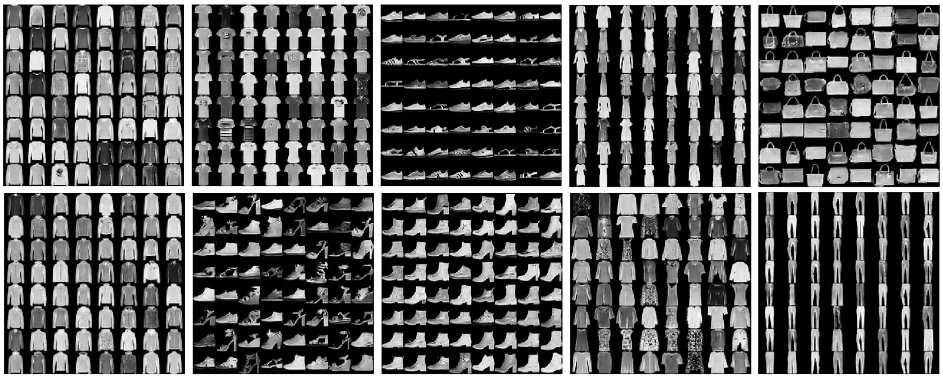


Figure 4.11: Generated images from each latent component of SLOGAN trained on the Fashion-MNIST dataset.

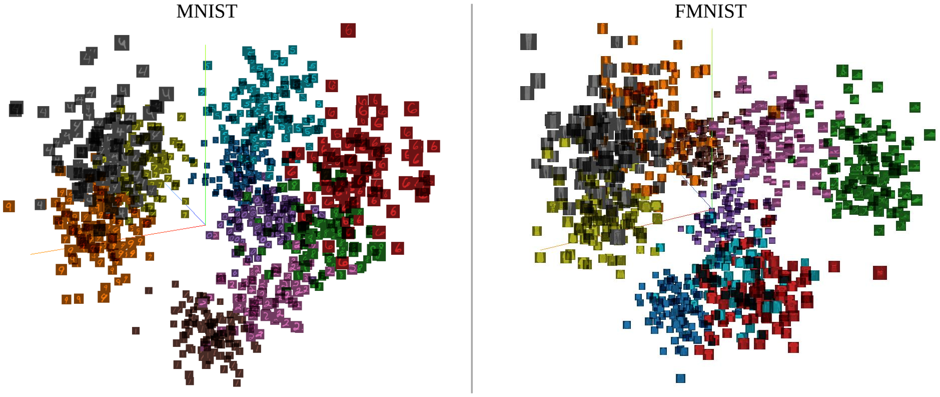


Figure 4.12: 3D PCA of the latent spaces of SLOGAN trained on the MNIST and Fashion-MNIST datasets.

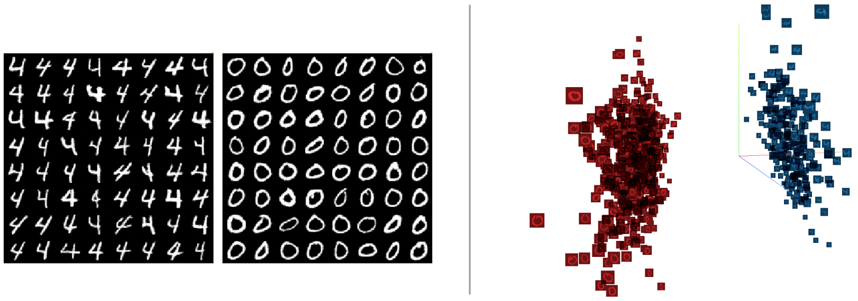


Figure 4.13: Generated images from each latent component and 3D PCA of the latent spaces of SLOGAN trained on the MNIST-2 (7:3) dataset.

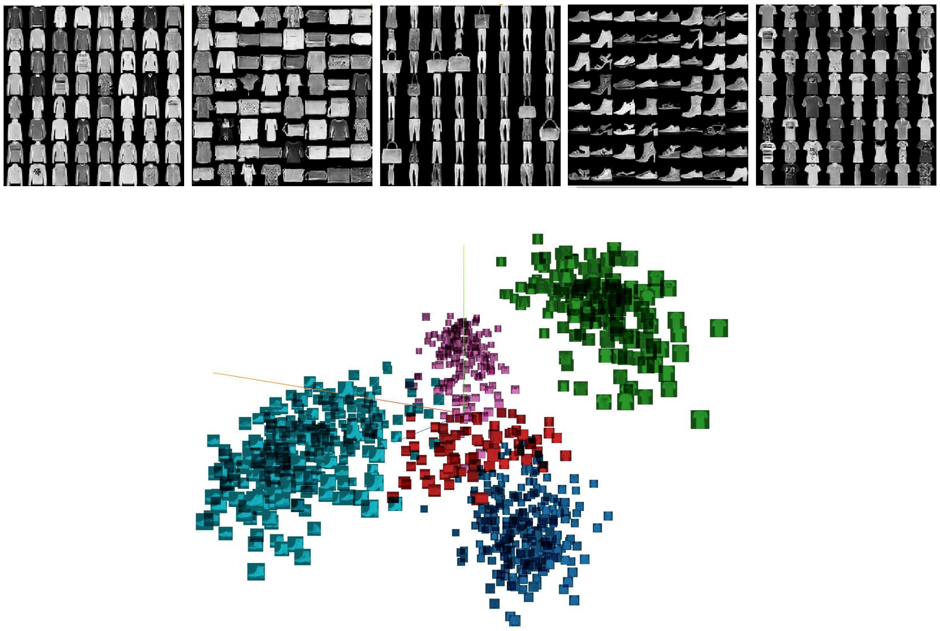
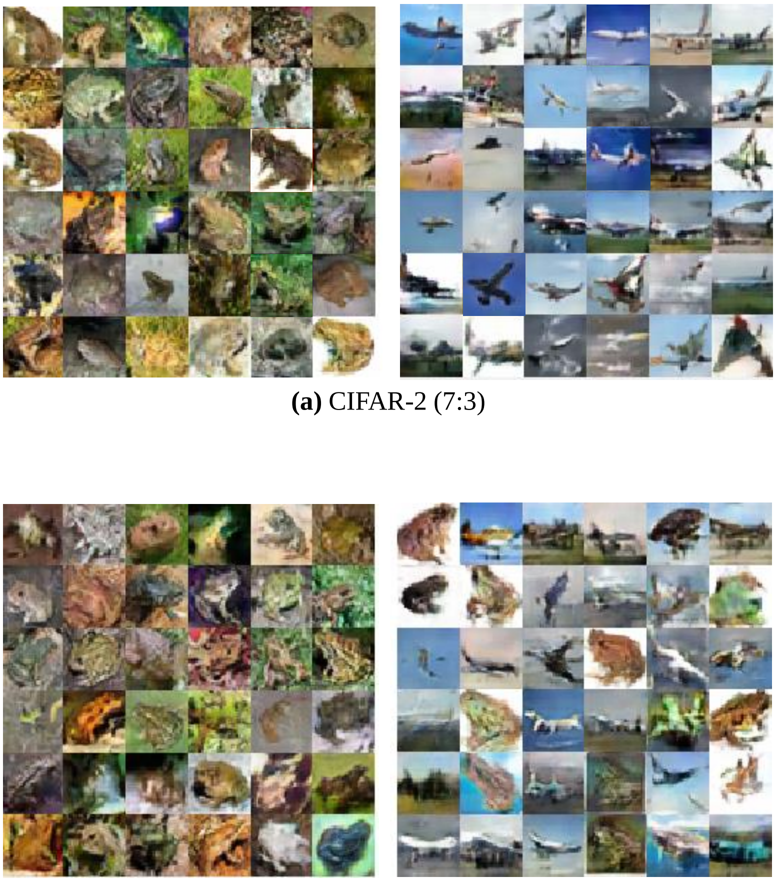


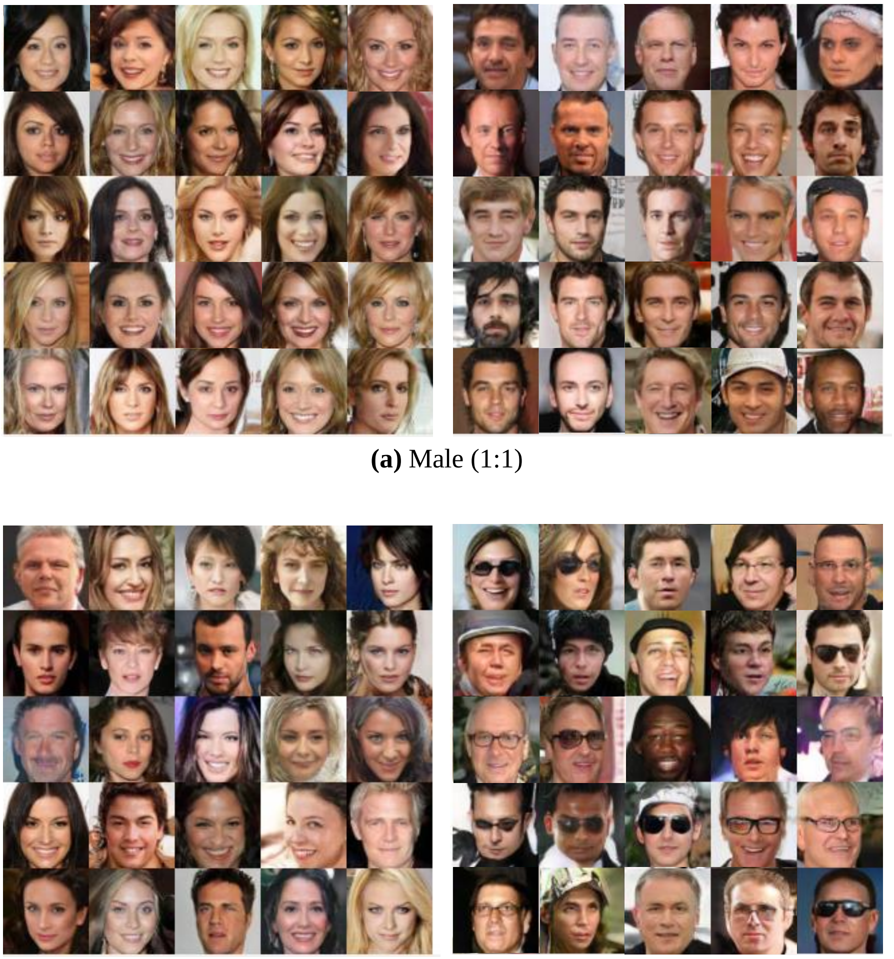
Figure 4.14: Generated images from each latent component and 3D PCA of the latent space of SLOGAN trained on the FMNIST-5 dataset.



(a) CIFAR-2 (7:3)

(b) CIFAR-2 (9:1)

Figure 4.15: Generated images from each latent component of SLOGAN trained on the (a) CIFAR-2 (7:3) and (b) CIFAR-2 (9:1) datasets.



(a) Male (1:1)

(b) Eyeglasses (14:1)

Figure 4.16: Generated images from each latent component of SLOGAN trained on the CelebA dataset. We used 30 probe data ((a) Female vs. Male, or (b) Faces without eyeglasses vs. Faces with eyeglasses) and mixup for each component.

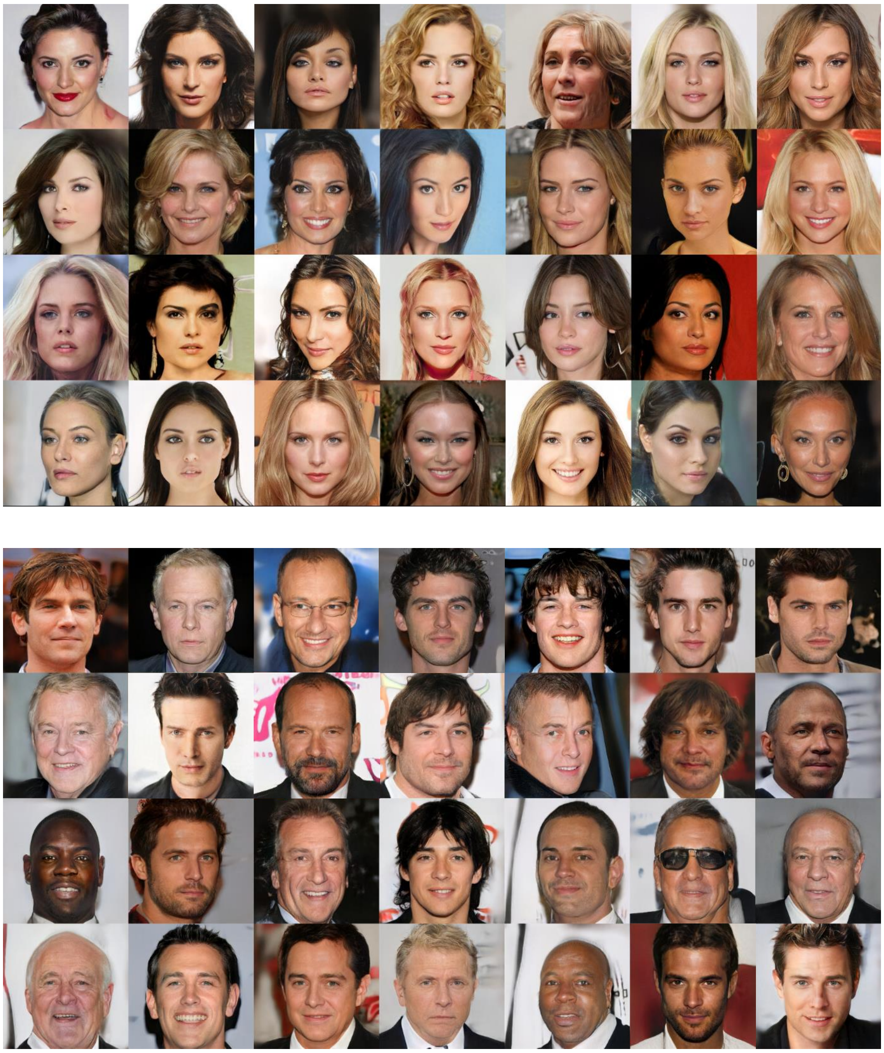
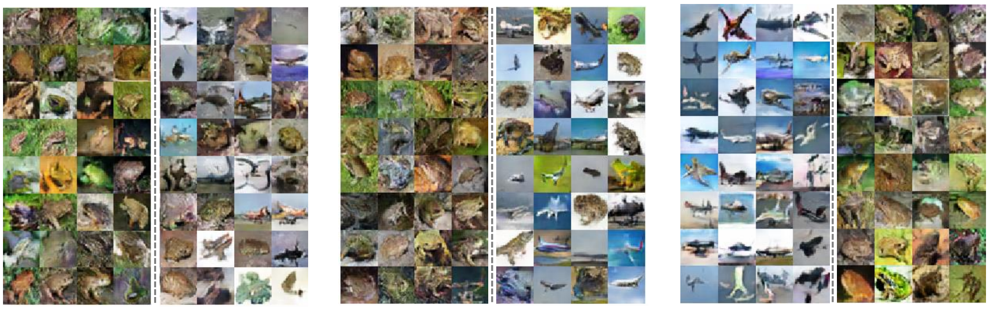
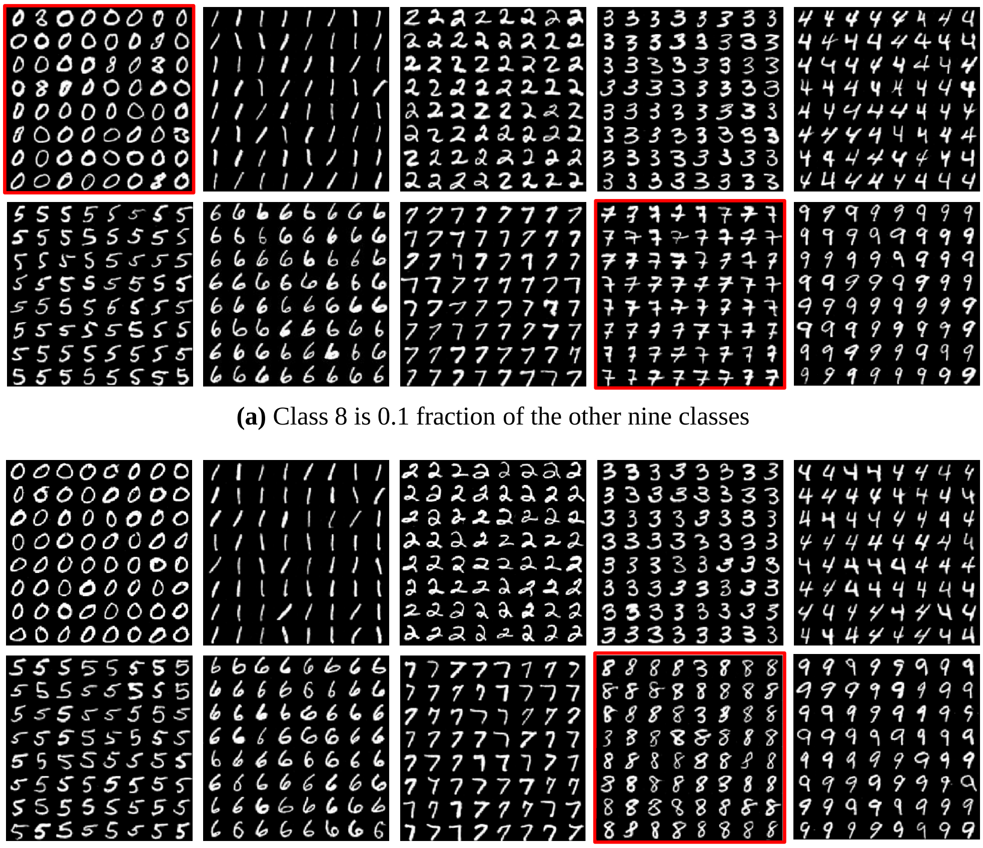


Figure 4.17: Generated images from each latent component of SLOGAN trained on the CelebA-HQ (256×256) dataset. We used 30 probe data (Female vs. Male) and mixup for each component.



(a) CD-GAN **(b)** PGMGAN **(c)** SLOGAN

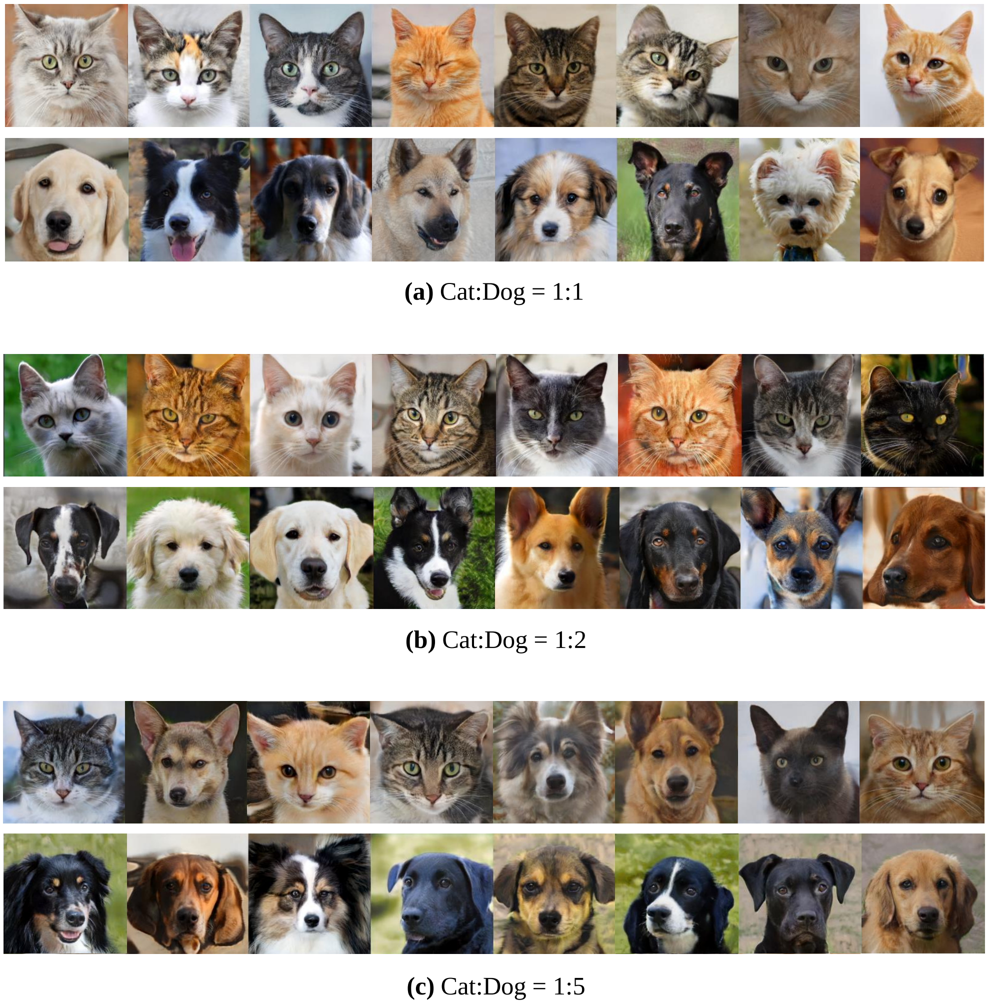
Figure 4.18: Generated images from the most recent methods including (a) CD-GAN, (b) PGMGAN, and (c) SLOGAN trained on the CIFAR-2 (7:3) dataset.



(a) Class 8 is 0.1 fraction of the other nine classes

(b) Class 8 is 0.2 fraction of the other nine classes

Figure 4.19: Generated images from each latent component of SLOGAN trained on the MNIST dataset where class 8 is very low fraction of the other nine classes.



(a) Cat:Dog = 1:1

(b) Cat:Dog = 1:2

(c) Cat:Dog = 1:5

Figure 4.20: Generated images from each latent component of SLOGAN trained on Cats and Dogs of the AFHQ (256×256) dataset with various imbalance ratios.

4.5 Appendix

4.5.1 Proofs

Gradient Identity for $\boldsymbol{\mu}_c$

Theorem 4.1. *Given an expected loss of the generator $\mathcal{L}$ and a loss function for a sample $\ell(\cdot) : \mathbb{R}^{d_z} \mapsto \mathbb{R}$, we assume ℓ is locally absolute continuous on almost every straight line (ACL) and continuous. Then, the following identity holds:*

$$\nabla_{\boldsymbol{\mu}_c} \mathcal{L} = \mathbb{E}_q \left[\delta(\mathbf{z})_c \pi_c \nabla_{\mathbf{z}} \ell(\mathbf{z}) \right] \tag{4.14}$$

Proof. To prove Theorem 4.1, the following lemma (Bonnet's theorem) is introduced.

Lemma 4.1 (Bonnet [17])**.** *Let $h(\mathbf{z}) : \mathbb{R}^d \mapsto \mathbb{R}$ be locally ACL and continuous. $q(\mathbf{z})$ is a multivariate Gaussian distribution $\mathcal{N}(\mathbf{z}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$. Then, the following identity holds:*

$$\nabla_{\boldsymbol{\mu}} \mathbb{E}_{q(\mathbf{z})} \left[h(\mathbf{z}) \right] = \mathbb{E}_{q(\mathbf{z})} \left[\nabla_{\mathbf{z}} h(\mathbf{z}) \right] \tag{4.15}$$

The proof of Lemma 4.1 is described by Theorem 3 of [98]. Using Lemma 4.1, we show that

$$\begin{aligned}
\nabla_{\boldsymbol{\mu}_c} \mathcal{L} = \nabla_{\boldsymbol{\mu}_c} \mathbb{E}_{q(\mathbf{z})} \left[\ell(\mathbf{z}) \right] = \nabla_{\boldsymbol{\mu}_c} \sum_{i=1}^{K} p(i) \mathbb{E}_{q(\mathbf{z}|i)} \left[\ell(\mathbf{z}) \right] && (4.16) \\
= p(c) \nabla_{\boldsymbol{\mu}_c} \mathbb{E}_{q(\mathbf{z}|c)} \left[\ell(\mathbf{z}) \right] = p(c) \mathbb{E}_{q(\mathbf{z}|c)} \left[\nabla_{\mathbf{z}} \ell(\mathbf{z}) \right] && (4.17) \\
= \int q(\mathbf{z}|c) p(c) \nabla_{\mathbf{z}} \ell(\mathbf{z}) d\mathbf{z} && (4.18) \\
= \int q(\mathbf{z}) \frac{q(\mathbf{z}|c) p(c)}{q(\mathbf{z})} \nabla_{\mathbf{z}} \ell(\mathbf{z}) d\mathbf{z} && (4.19) \\
= \int q(\mathbf{z}) \delta(\mathbf{z})_c \pi_c \nabla_{\mathbf{z}} \ell(\mathbf{z}) d\mathbf{z} && (4.20) \\
= \mathbb{E}_{q(\mathbf{z})} \left[\delta(\mathbf{z})_c \pi_c \nabla_{\mathbf{z}} \ell(\mathbf{z}) \right] && (4.21)
\end{aligned}$$

□

First-order Gradient Identity for Σ_c

Theorem 4.2. *Under the same assumptions from Theorem 4.1 and assuming that* $\mathbb{E}_q[\ell(\mathbf{z})]$ *is well-defined, we have the following gradient identity:*

$$\nabla_{\Sigma_c}\mathcal{L} = \frac{1}{2}\mathbb{E}_q\left[\delta(\mathbf{z})_c\pi_c\Sigma_c^{-1}\left(\mathbf{z}-\boldsymbol{\mu}_c\right)\nabla_{\mathbf{z}}^T\ell(\mathbf{z})\right] \tag{4.22}$$

Proof. In order to prove Theorem 4.2, we introduce the following lemma (Price's theorem).

Lemma 4.2 (Price [128])**.** *Let* $h(\mathbf{z}) : \mathbb{R}^d \mapsto \mathbb{R}$ *be locally ACL and continuous. We further assume that* $\mathbb{E}_q\left[h(\mathbf{z})\right]$ *is well-defined. Then, the following identity holds:*

$$\nabla_{\boldsymbol{\Sigma}}\mathbb{E}_{q(\mathbf{z})}\left[h(\mathbf{z})\right] = \frac{1}{2}\mathbb{E}_{q(\mathbf{z})}\left[\boldsymbol{\Sigma}^{-1}(\mathbf{z}-\boldsymbol{\mu})\nabla_{\mathbf{z}}^T h(\mathbf{z})\right] = \frac{1}{2}\mathbb{E}_{q(\mathbf{z})}\left[\nabla_{\mathbf{z}}^2 h(\mathbf{z})\right] \tag{4.23}$$

The proof of Lemma 4.2 is presented in Theorem 4 of [98]. The rest of the proof is similar with the proof of Theorem 4.1. Using the first-order gradient identity of Lemma 4.2, we get

$$\begin{aligned}
\nabla_{\Sigma_c}\mathcal{L} &= \nabla_{\Sigma_c}\mathbb{E}_{q(\mathbf{z})}\left[\ell(\mathbf{z})\right] = \nabla_{\Sigma_c}\sum_{i=1}^{K}p(i)\mathbb{E}_{q(\mathbf{z}|i)}\left[\ell(\mathbf{z})\right] && (4.24)\\
&= p(c)\nabla_{\Sigma_c}\mathbb{E}_{q(\mathbf{z}|c)}\left[\ell(\mathbf{z})\right] = \frac{1}{2}p(c)\mathbb{E}_{q(\mathbf{z}|c)}\left[\Sigma_c^{-1}(\mathbf{z}-\boldsymbol{\mu}_c)\nabla_{\mathbf{z}}^T\ell(\mathbf{z})\right] && (4.25)\\
&= \frac{1}{2}\int q(\mathbf{z}|c)p(c)\Sigma_c^{-1}(\mathbf{z}-\boldsymbol{\mu}_c)\nabla_{\mathbf{z}}^T\ell(\mathbf{z})d\mathbf{z} && (4.26)\\
&= \frac{1}{2}\int q(\mathbf{z})\frac{q(\mathbf{z}|c)p(c)}{q(\mathbf{z})}\Sigma_c^{-1}(\mathbf{z}-\boldsymbol{\mu}_c)\nabla_{\mathbf{z}}^T\ell(\mathbf{z})d\mathbf{z} && (4.27)\\
&= \frac{1}{2}\int q(\mathbf{z})\delta(\mathbf{z})_c\pi_c\Sigma_c^{-1}(\mathbf{z}-\boldsymbol{\mu}_c)\nabla_{\mathbf{z}}^T\ell(\mathbf{z})d\mathbf{z} && (4.28)\\
&= \frac{1}{2}\mathbb{E}_{q(\mathbf{z})}\left[\delta(\mathbf{z})_c\pi_c\Sigma_c^{-1}(\mathbf{z}-\boldsymbol{\mu}_c)\nabla_{\mathbf{z}}^T\ell(\mathbf{z})\right] && (4.29)
\end{aligned}$$

□

Ensuring Positive-definiteness of Σ_c

Theorem 4.3. *The updated covariance matrix* $\Sigma_c' = \Sigma_c + \gamma\Delta\Sigma_c'$ *with the modified update rule specified in Equation 4.5 is positive-definite if* Σ_c *is positive-definite.*

Proof. Because Σ_c is symmetric and positive-definite, we can decompose $\Sigma_c = \mathrm{L}\mathrm{L}^T$ using Cholesky decomposition, where L is the lower triangular matrix. Then, we can prove the positive definiteness of the updated covariance matrix as follows:

$$\begin{aligned}
\Sigma_c' &= \Sigma_c + \gamma\Delta\Sigma_c' && (4.30)\\
&= \Sigma_c + \gamma(\Delta\Sigma_c + \frac{\gamma}{2}\Delta\Sigma_c\Sigma_c^{-1}\Delta\Sigma_c) && (4.31)\\
&= \Sigma_c + \gamma\Delta\Sigma_c + \frac{\gamma^2}{2}\Delta\Sigma_c\Sigma_c^{-1}\Delta\Sigma_c && (4.32)\\
&= \frac{1}{2}\left(\Sigma_c + (\mathrm{L} + \gamma\Delta\Sigma_c\mathrm{L}^{-T})(\mathrm{L}^T + \gamma\mathrm{L}^{-1}\Delta\Sigma_c)\right) && (4.33)\\
& && (4.34)
\end{aligned}$$

Let us define $\mathrm{U} := \mathrm{L}^T + \gamma\mathrm{L}^{-1}\Delta\Sigma_c$. Then, we have the following:

$$\Sigma_c' = \frac{1}{2}\left(\Sigma_c + \mathrm{U}^T\mathrm{U}\right) \succ 0 \qquad (4.35)$$

where both Σ_c and $\mathrm{U}^T\mathrm{U}$ are positive-definite, concluding the proof. □

Gradient Identity for ρ_c

Theorem 4.4. *Let* ρ_c *be a mixing coefficient parameter, and the following gradient identity holds:*

$$\nabla_{\rho_c}\mathcal{L} = \mathbb{E}_q\left[\pi_c\left(\delta(\mathbf{z})_c - 1\right)\ell(\mathbf{z})\right] \qquad (4.36)$$

Proof. The gradient of the latent distribution with respect to the mixing coefficient parameter is derived as follows:

$$\nabla_{\rho_c} q(\mathbf{z}) = \nabla_{\rho_c} \sum_{i=1}^{K} \mathrm{softmax}(\rho_i) q(\mathbf{z}|i) = \pi_c \left(q(\mathbf{z}|c) - \sum_{i=1}^{K} \pi_i q(\mathbf{z}|i) \right) \tag{4.37}$$

where $\mathrm{softmax}(\cdot)$ denotes the softmax function (e.g., $p(i) = \pi_i = \mathrm{softmax}(\rho_i)$). Using the above equation, we have

$$\begin{aligned} \nabla_{\rho_c} \mathcal{L} = \nabla_{\rho_c} \mathbb{E}_{q(\mathbf{z})} \left[\ell(\mathbf{z}) \right] &= \int \nabla_{\rho_c} q(\mathbf{z}) \ell(\mathbf{z}) d\mathbf{z} &(4.38)\\ &= \int \pi_c \left(q(\mathbf{z}|c) - \sum_{i=1}^{K} \pi_i q(\mathbf{z}|i) \right) \ell(\mathbf{z}) d\mathbf{z} &(4.39)\\ &= \int \pi_c q(\mathbf{z}) \left(\frac{q(\mathbf{z}|c)}{q(\mathbf{z})} - \sum_{i=1}^{K} \pi_i \frac{q(\mathbf{z}|i)}{q(\mathbf{z})} \right) \ell(\mathbf{z}) d\mathbf{z} &(4.40)\\ &= \int q(\mathbf{z}) \pi_c \left(\delta(\mathbf{z})_c - \sum_{i=1}^{K} \pi_i \delta(\mathbf{z})_i \right) \ell(\mathbf{z}) d\mathbf{z} &(4.41) \end{aligned}$$

Here, $\sum_{i=1}^{K} \pi_i \delta(\mathbf{z})_i = \sum_{i=1}^{K} \frac{p(i)q(\mathbf{z}|i)}{q(\mathbf{z})} = 1$. Plugging this back into Equation 4.41, we obtain

$$\begin{aligned} \nabla_{\rho_c} \mathcal{L} &= \int q(\mathbf{z}) \pi_c \left(\delta(\mathbf{z})_c - 1 \right) \ell(\mathbf{z}) d\mathbf{z} &(4.42)\\ &= \mathbb{E}_{q(\mathbf{z})} \left[\pi_c \left(\delta(\mathbf{z})_c - 1 \right) \ell(\mathbf{z}) \right] &(4.43) \end{aligned}$$

□

Discussion on assumptions for gradient identities

In this section, we discuss the assumptions of Theorems 4.1 and 4.2. We investigate whether the loss function used to train the Gaussian mixture parameters is approximately locally ACL. To begin with, we note that G, D, and E are neural networks with (leaky)

ReLU activations, and their Lipschitz constants can be expressed as follows [113]:

$$\|f\|_{\mathrm{Lip}} \leq \|\mathbf{h}_L \to W^{L+1}\mathbf{h}_L\|_{\mathrm{Lip}} \cdot \cancel{\|a_L\|_{\mathrm{Lip}}} \cdot \|\mathbf{h}_{L-1} \to W^L \mathbf{h}_{L-1}\|_{\mathrm{Lip}} \tag{4.44}$$

$$\dots \cancel{\|a_1\|_{\mathrm{Lip}}} \cdot \|\mathbf{h}_0 \to W^L \mathbf{h}_0\|_{\mathrm{Lip}} = \prod_{l=1}^{L+1} \|\mathbf{h}_{l-1} \to W^l \mathbf{h}_{l-1}\|_{\mathrm{Lip}} \tag{4.45}$$

$$= \prod_{l=1}^{L+1} \sigma(W^l) \tag{4.46}$$

where $\|\cdot\|_{\mathrm{Lip}}$ is Lipschitz constant of a function, $\sigma(\cdot)$ is Lipschitz (spectral) norm of a matrix, W^l denotes the weight matrix of the l-th layer, and $\mathbf{h}_0$ and $\mathbf{h}_l$ are the input vector and the output vector of the l-th layer, respectively. If the Lipschitz (spectral) norms of weight matrices are finite, D, G, and E are Lipschitz continuous. Each component regularizes the Lipschitz norms of weight matrices as the following remark.

Remark 1. *Each network (G, D, and E) is approximately regularized for Lipschitz continuity to approximate the assumptions of being locally ACL and continuous.*

- *G regularizes the Lipschitz norm of the model weights through batch normalization. Santurkar et al. [138] demonstrated that batch normalization improves the Lipschitzness of the loss and the gradients of deep models.*

- *D regularizes the Lipschitz norms of the model weights through adversarial Lipschitz regularization [152]. Although the entire model function may not become Lipschitz continuous, this makes the model function approximately Lipschitz continuous around the samples to which it is applied.*

- *E regularizes the Lipschitz norms of the model weights by adding L2-regularization to the loss. Minimizing the L2 norm of weights smoothes the model function, which is expected to indirectly contribute to the regularization of the Lipschitz norms of the model weights.*

Now, we will examine the Lipschitz continuity of the adversarial loss computed from the output of D and the U2C loss computed from the output of E. The adversarial loss is given by $\ell_{\text{adv}} = -D\left(G(\mathbf{z}^i)\right)$ and represents the composition of functions D and G, making it Lipschitz continuous.

The U2C loss is defined as $\ell_{\text{U2C}} = -\log \frac{\exp(\cos\theta_{ii})}{\frac{1}{B}\sum_{j=1}^{B}\exp(\cos\theta_{ij})}$, where the numerator involves a cosine function and the denominator consists of a log-sum-exponential function. First, we verify that the cosine function is Lipschitz continuous with respect to two variables x and y as follows:

$$|\cos x - \cos y| = \left|-2\sin\left(\frac{x+y}{2}\right)\sin\left(\frac{x-y}{2}\right)\right| \leq 2\left|\sin\left(\frac{x-y}{2}\right)\right| \tag{4.47}$$

$$\leq 2\left|\frac{x-y}{2}\right| = |x-y| \tag{4.48}$$

The inequality in Equation 4.48 is derived using the property of the sine function ($|\sin x| \leq |x|$). Additionally, the log-sum-exponential function is Lipschitz continuous as follows:

$$|f(x) - f(y)| = |\nabla f(c)\cdot(x-y)| \leq \|x-y\|_2 \cdot \|\nabla f(c)\|_2 \tag{4.49}$$

$$\leq \|x-y\|_2 \cdot \|\nabla f(c)\|_1 = \|x-y\|_2 \tag{4.50}$$

The inequality in Equation 4.49 is derived from Hölder's inequality, and the inequality and the equality in Equation 4.50 is derived from the following Cauchy-Schwarz inequality and the property of the log-sum-exponential function, respectively:

$$\|x\|_2^2 = \sum_i x_i^2 \leq (\sum_i |x_i|)(\sum_i |x_i|) = \|x\|_1^2 \tag{4.51}$$

$$\|\nabla f(c)\|_1 = \sum_{i=1}^{n} f_{x_i}(c) = 1 \tag{4.52}$$

where $f(x) = \log\sum_{i=1}^{n} e^{x_i}$ and $f_{x_i}(x) = \frac{e^{x_i}}{e^{x_1}+\cdots+e^{x_n}}$ is the i-th partial derivative of f. Therefore, loss functions in SLOGAN approximately are approximately Lipschitz continuous. Since Lipschitz continuous function is locally ACL and continuous [98], loss functions in SLOGAN approximately satisfy the assumptions for gradient identities.

However, regarding the scalability of the model, we should note the following remark.

Remark 2. *It is observed that as the dimensions of each layer in neural networks increase, the Lipschitz constant tends to increase [45]. Without careful regularization, the model training can become unstable.*

4.5.2 Implementation details

Additive angular margin

To enhance the discriminative power of U2C loss, we adopted the additive angular margin [35] as follows:

$$\ell_{\mathrm{U2C}}(\mathbf{z}^i) = -\log \frac{\exp(s \cdot \cos(\theta_{ii} + m))}{\frac{1}{B}\{\exp(s \cdot \cos(\theta_{ii} + m)) + \sum_{j \neq i} \exp(s \cdot \cos\theta_{ij})\}} \tag{4.53}$$

where s denotes the feature scale, and m is the angular margin. The feature scale m and the coefficient of U2C loss λ are linearly decayed to 0 during training, so that SLOGAN can focus more on the adversarial loss as training progresses.

DeLiGAN+

Among the existing unsupervised conditional GANs, DeLiGAN lacks an encoder network. Therefore, for a fair comparison, we added an encoder network and named it DeLiGAN+. We set the output dimension of the encoder to equal the number of mixture components of the latent distribution. For the i-th example in the batch, when the c_i-th mixture component of the latent distribution is selected, DeLiGAN+ is learned through the following objective:

$$\min_{G,E,\mu_c,\sigma_c} \max_{D} \frac{1}{B} \sum_{i=1}^{B} \left[D(\mathbf{x}^i) - D(G(\mathbf{z}^i, \mathbf{c}_i)) - \lambda_{CE}\, \mathbf{c}_i^T \log E(G(\mathbf{z}^i, \mathbf{c}_i)) \right] \tag{4.54}$$

where $\mathbf{c}_i$ is the one-hot vector corresponding to c_i and λ_{CE} is the coefficient of the cross entropy loss. We set λ_{CE} to 10 in the experiments.

Evaluation Metric

Cluster assignment We do not use clustering purity which is an evaluation metric for cluster assignment. To compute the clustering purity, the most frequent class in the cluster is obtained, and the ratio of the data points belonging to the class is calculated. However, if the attributes in the data are imbalanced, multiple clusters can be assigned to a single class in duplicate, and this high clustering purity misleads the results. Therefore, we utilized the normalized mutual information (NMI) implemented in scikit-learn[2].

Unconditional generation FID has the advantage of considering not only sample quality but also diversity, whereas Inception score (IS) cannot assess the diversity properly because IS does not compare generated samples with real samples [140]. Therefore, we used FID as the evaluation metric for unsupervised generation.

Unsupervised conditional generation If attributes in data are severely imbalanced, FID does not increase (deteriorate) considerably even if the model does not generate data containing the minority attributes. Therefore, the FID cannot accurately measure the unsupervised conditional generation performance for data with severely imbalanced attributes. We introduce ICFID to evaluate the performance of unsupervised conditional generation. When calculating ICFID, multiple clusters cannot be assigned to a single class in duplicate. Therefore, if data of a single class are generated from multiple discrete latent variables or modes, the model shows high (bad) ICFID.

General Settings and Environments

For simplicity, we denote the learning rate of G as η and the learning rate of $\mathbf{\Sigma}$ as γ. Throughout the experiments, we set the learning rate of E to η, and D to 4η using

[2] https://github.com/scikit-learn/scikit-learn/blob/15a949460/sklearn/metrics/cluster/_supervised.py

Table 4.7: SLOGAN architecture used for the synthetic dataset

G	D	E
$\mathbf{z} \in \mathbb{R}^{64}$	$\mathbf{x} \in \mathbb{R}^{2}$	$\mathbf{x} \in \mathbb{R}^{2}$
Linear 128 + BN + ReLU	Linear 128 + LReLU	Linear 128 + SN + LReLU
Linear 128 + BN + ReLU	Linear 128 + LReLU	Linear 128 + SN + LReLU
Linear 2 + Tanh	Linear 1	Linear 64 + SN

the two-timescale update rule (TTUR) [66]. We set the learning rate of $\boldsymbol{\mu}$ to 10γ, and the learning rate of $\boldsymbol{\rho}$ to γ. We set B to 64 and the number of training steps to 100k. To stabilize discriminator learning, we used Lipschitz penalty [126] for the synthetic, MNIST, FMNIST, and 10x_73k datasets, and adversarial Lipschitz regularization [152] for the CIFAR-10 and CelebA datasets. We repeated each experiment 3 times and reported the means and standard deviations of model performances. Hyperparameters are determined by a grid search. We used the Adam optimizer [82] for training D, G, and E, and a gradient descent optimizer for training $\boldsymbol{\mu}$, $\boldsymbol{\Sigma}$, and $\boldsymbol{\rho}$. The experiments herein were conducted on a machine equipped with an Intel Xeon Gold 6242 CPU and an NVIDIA Quadro RTX 8000 GPU. The code is implemented in Python 3.7 and Tensorflow 1.14 [1].

Preprocessing and architecture

Synthetic Dataset For the synthetic dataset, we first set the mean of eight 2-dimensional Gaussian distributions as $(0, 2)$, $(\sqrt{2}, \sqrt{2})$, $(2, 0)$, $(\sqrt{2}, -\sqrt{2})$, $(0, -2)$, $(-\sqrt{2}, -\sqrt{2})$, $(-2, 0)$, and $(-\sqrt{2}, \sqrt{2})$, and the variance as $0.01I$. The number of data sampled from the Gaussian distributions was set to 5,000, 5,000, 5,000, 5,000, 15,000, 15,000, 15,000, and 15,000. We scaled a total of 80,000 data points to a range between -1 and 1. Table 4.7 shows the network architectures of SLOGAN used for the synthetic dataset. Linear n denotes a fully-connected layer with n output units. BN and SN denote batch normalization and spectral normalization, respectively. LReLU denotes the leaky ReLU. We set $\lambda = 4$, $\eta = 0.001$, $\gamma = 0.01$, $s = 2$, and $m = 0.5$.

Table 4.8: SLOGAN architecture used for the MNIST and FMNIST datasets

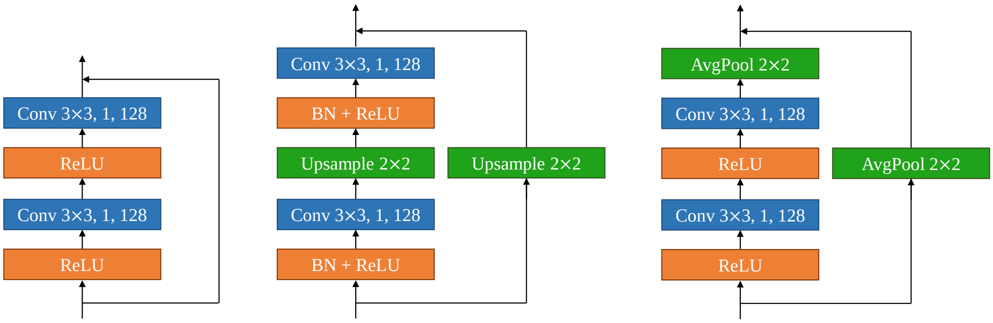
G	D	E
$\mathbf{z} \in \mathbb{R}^{1\times1\times64}$	$\mathbf{x} \in \mathbb{R}^{28\times28\times1}$	$\mathbf{x} \in \mathbb{R}^{28\times28\times1}$
Deconv 1×1, 1, 1024 + BN + ReLU	Conv 4×4, 2, 64 + LReLU	Conv 4×4, 2, 64 + LReLU
Deconv 7×7, 1, 128 + BN + ReLU	Conv 4×4, 2, 64 + LReLU	Conv 4×4, 2, 64 + LReLU
Deconv 4×4, 2, 64 + BN + ReLU	Conv 7×7, 1, 1024 + LReLU	Conv 7×7, 1, 1024 + LReLU
Deconv 4×4, 2, 1 + Sigmoid	Conv 1×1, 1, 1	Conv 1×1, 1, 64

MNIST and Fashion-MNIST Datasets The MNIST dataset [93] consists of handwritten digits, and the Fashion-MNIST (FMNIST) dataset [169] is comprised of fashion products. Both the MNIST and Fashion-MNIST datasets have 60,000 training and 10,000 test 28×28 grayscale images. Each pixel was scaled to a range of $0-1$. The datasets consist of 10 classes, and the number of data points per class is balanced. Table 4.8 shows the network architectures of SLOGAN used for the MNIST and FMNIST datasets. Conv $k \times k$, s, n denotes a convolutional network with n feature maps, filter size $k \times k$, and stride s. Deconv $k \times k$, s, n denotes a deconvolutional network with n feature maps, filter size $k \times k$, and stride s. For the MNIST dataset, we set $\lambda = 10$, $\eta = 0.0001$, $\gamma = 0.002$, $s = 8$, and $m = 0.5$. For MNIST-2, we set $\lambda = 4$, $\eta = 0.0001$, $\gamma = 0.002$, $s = 4$, and $m = 0.5$. For the FMNIST dataset, we set $\lambda = 10$, $\eta = 0.0001$, $\gamma = 0.001$, $s = 1$, and $m = 0$. For FMNIST-5, we set $\lambda = 1$, $\eta = 0.0002$, $\gamma = 0.004$, $s = 4$, and $m = 0.5$.

10x_73k Dataset The 10x_73k dataset [176] consists of 73,233 720-dimensional vectors, which are obtained from RNA transcript counts, and has eight cell types (classes). The number of data points per cell type is 10,085, 2,612, 9,232, 8,385, 10,224, 11,953, 10,479, and 10,263. We converted each element to logscale (i.e., $\log_2(x+1)$) and scaled each element to a range between 0 and 1. Table 4.9 shows the network architectures of SLOGAN used for the 10x_73k dataset. We set $\lambda = 10$, $\eta = 0.0001$, $\gamma = 0.004$, $s = 4$, and $m = 0$.

Table 4.9: SLOGAN architecture used for the 10x_73k dataset

G	D	E
$\mathbf{z} \in \mathbb{R}^{64}$	$\mathbf{x} \in \mathbb{R}^{2}$	$\mathbf{x} \in \mathbb{R}^{2}$
Linear 256 + LReLU	Linear 256 + LReLU	Linear 256 + SN + LReLU
Linear 256 + LReLU	Linear 256 + LReLU	Linear 256 + SN + LReLU
Linear 720	Linear 1	Linear 64 + SN



(a) Resblock (b) Resblock Up (c) Resblock Down

Figure 4.21: Resblock architectures used for colored image datasets.

Table 4.10: SLOGAN architecture used for the CIFAR datasets

G	D	E
$\mathbf{z} \in \mathbb{R}^{128}$	$\mathbf{x} \in \mathbb{R}^{32\times32\times3}$	$\mathbf{x} \in \mathbb{R}^{32\times32\times3}$
Linear 2048 + Reshape 4, 4, 128	Resblock Down	Resblock Down
Resblock Up	Resblock Down	Resblock Down
Resblock Up	Resblock	Resblock
Resblock Up	Resblock	Resblock
BN + ReLU	ReLU + GlobalAvgPool	ReLU + GlobalAvgPool
Conv 3×3, 1, 3 + Tanh	Linear 1	Linear 128

CIFAR-10 Dataset The CIFAR-10 [88] dataset comprises 50,000 training and 10,000 test 32×32 color images. Each pixel was scaled to a range of -1 to 1. The number of data points per class is balanced. Figure 4.21 and Table 4.10 show the network architectures of residual blocks and SLOGAN used for the CIFAR datasets. AvgPool and GlobalAvgPool denote the average pooling and global average pooling layers, respectively. For the CIFAR-10, CIFAR-2, CIFAR-2 (7:3), and CIFAR-2 (9:1) datasets, we set $\lambda = 1$, $\eta = 0.0001$, $\gamma = 0.002$, $s = 4$, and $m = 0.5$.

Table 4.11: SLOGAN architecture used for the CelebA dataset

G	D	E
$\mathbf{z} \in \mathbb{R}^{128}$	$\mathbf{x} \in \mathbb{R}^{64 \times 64 \times 3}$	$\mathbf{x} \in \mathbb{R}^{64 \times 64 \times 3}$
Linear 8192 + Reshape 8, 8, 128	Resblock Down	Resblock Down
Resblock Up	Resblock Down	Resblock Down
Resblock Up	Resblock Down	Resblock Down
Resblock Up	Resblock	Resblock
BN + ReLU	ReLU + GlobalAvgPool	ReLU + GlobalAvgPool
Conv 3×3, 1, 3 + Tanh	Linear 1	Linear 128

CelebA Dataset The CelebA dataset [101] consists of 202,599 face attributes. We cropped the face part of each image to 140×140 pixels, resized it to 64×64 pixels, and scaled it to a range between -1 and 1. The imbalanced ratio is different for each attribute, and the imbalanced ratios of male and eyeglasses used in the experiment are 1:1 and 14:1, respectively. Table 4.11 lists the network architectures of SLOGAN used for the CelebA dataset. We set $\lambda = 1$, $\eta = 0.0002$, $\gamma = 0.0006$, $s = 4$, and $m = 0.5$.

CelebA-HQ Dataset The CelebA-HQ dataset [78] consists of 30,000 face attributes. We resized each image to 128×128 and 256×256 pixels, and scaled it to a range between -1 and 1. The imbalance ratio of male used in the experiment was 1.7:1. We used StyleGAN2 [80] architecture with DiffAugment[3] and applied implicit reparameterization to the input space of the mapping network. We set $\lambda = 1$, $\eta = 0.002$, $\gamma = 0.0006$, $s = 1$, and $m = 0$.

AFHQ Dataset The AFHQ dataset [29] consists of 15,000 high-quality animal faces. We used cats and dogs, and there are about 5,000 images each in the dataset. We resized each image to 256×256 pixels, and scaled it to a range between -1 and 1. We set the imbalance ratios of cats and dogs to 1:1, 1:2, and 1:5 by reducing the number of cats in the training dataset. We used the same model architecture and hyperparameters as for the CelebA-HQ dataset.

[3] https://github.com/mit-han-lab/data-efficient-gans/tree/master/DiffAugment-stylegan2

Code Availability

Code is available at https://github.com/shinyflight/SLOGAN

4.6 Summary

In this chapter, we have proposed a method called SLOGAN to generate data conditioned on learned attributes on real-world datasets with balanced or imbalanced attributes. We derive implicit reparameterization for the parameters of the latent distribution. We then proposed a GAN framework and unsupervised conditional contrastive loss (U2C loss).

We verified that SLOGAN achieved state-of-the-art unsupervised conditional generation performance. In addition, a small amount of probe data helps SLOGAN control attributes.

SLOGAN requires specifying the number of mixture components K before training. In future work, we will consider a principled method to learn the number and hierarchy of attributes in real-world data. In addition, improving the quality of samples with minority attributes is an important avenue for future research on unsupervised conditional GANs.

Chapter 5

Applications of DGMs for EHRs

5.1 Prediction of Preclinical Alzheimer's Disease

5.1.1 Introduction

The possibility of preventing the development of Alzheimer's disease (AD) is receiving increasing attention, especially because few effective disease-modifying treatments (DMTs) have been developed [31, 32]. Amyloid beta (Aβ) deposition is used as a biomarker in prevention since it is a measure of the pathophysiological changes that take place in the preclinical stage of the disease [71, 115]. Aβ deposition can be measured directly using positron emission tomography (PET) or a lumbar puncture; but this is inappropriate for cognitively normal (CN) people because it is costly, time-consuming, and involves exposure to radiation or considerable pain. Proxy measures can be obtained relatively cheaply and safely using structural magnetic resonance imaging (MRI) [5, 14] or tests of the cognitive function [58, 86] are relatively inexpensive and safe, but the relationships between these measures and the extent of Aβ deposition are complicated.

Deep learning (DL) finds the complicated relationships between features and outcomes to predict outcomes from given features [94, 107]. If the outcome is a dichotomous variable, then deep learning constructs a non-linear decision boundary that

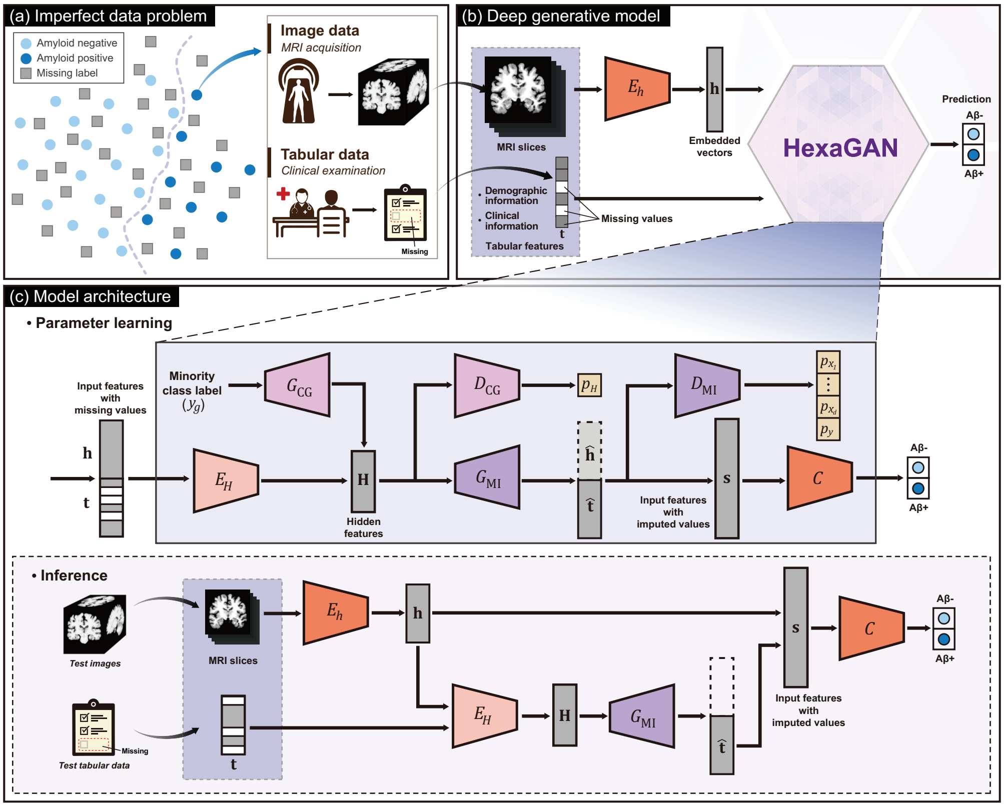


Figure 5.1: Methodological overview. (a) We aim to predict amyloid positivity from structural magnetic resonance imaging (MRI) scans, demographic information, and cognitive scores from cognitively normal (CN) individuals available in the Alzheimer's disease neuroimaging initiative (ADNI) database. In its raw state, this dataset is imperfect to be fed directly into a classifier. It includes observations with missing labels; it has missing values in clinical information; and amyloid negative (A$\beta-$) individuals outnumber amyloid positive (A$\beta+$) individuals. (b) We address these real-world problems simultaneously by enhancing the HexaGAN framework. Our model is able to deal with high-dimensional MRI scans because it has an additional encoder (E_h) to extract vectors of reduced dimensionality ($\mathbf{h}$). (c) The generators (G_{CG}, G_{MI}), discriminators (D_{CG}, D_{MI}), encoders (E_h, E_H), and classifier (C) are designed to learn the model from dataset having real-world problems. For inference, our model was designed to respond to scenarios where test features also have missing values or even when all tabular features are missing, using MRI scans alone to predict amyloid positivity.

separates data with one outcome from those with the other. In clinical applications, however, the dataset used for training is often imperfect: some features or outcome values are missing, or the number of examples with each outcome is very different. It can be caused by the cost of clinical tests, the cost of physician diagnosis, or the prevalence of diseases/the rarity of hospital visits by healthy people. These problems can hinder the generalization of the model by reducing the amount of training data or introducing biases for specific classes. In addition, different clinicians may diagnose diseases based on different sets of features or labeling criteria, which restricts the use of observations in the new dataset and makes the trained model less efficient or even useless in terms of model utility. Furthermore, the non-linearity of deep neural networks makes it difficult for clinicians to interpret the model's predictions and hence to explain them to patients. Addressing these issues is critical to improving the effectiveness and generalizability of the model.

In this chapter, we used a deep neural network to predict whether CN individuals are in the preclinical stage (or amyloid positive CN individuals) on the basis of data from structural MRI scans, demographic information, and clinical scores. Our model can make accurate predictions, and its development also embodies three significant steps toward real-world clinical application. Firstly, we trained our model on a dataset with missing features and labels, and with imbalanced classes: some cognitive scores have missing values, amyloid positivities are missing for some participants, and the size of the Aβ+ group is different from that of the A$\beta-$ group (Figure 5.1(a) and 5.1(b)). We addressed these problems by adopting the HexaGAN framework presented in Chapter 3 (Figure 5.1(c)). Secondly, we dealt with a situation in which a trained model is required to operate on data collected from different hospitals, in which feature sets or diagnostic criteria may be different, and constructed an appropriate prediction scheme. In this chapter, we show that HexaGAN is able to predict amyloid positivity even if the features obtained in testing do not perfectly match those used in model training. Thirdly, using explainable artificial intelligence (XAI) techniques, we determined discriminative regions and variables which represent the features that are important in the prediction of early amyloid pathology. These are determined at the population-level and thus provide physicians and patients with the reliability of the model's predictive ability. Therefore, our considerations for clinical implementation ranging from model architecture to application showed that our model can be successfully used in real-world situations.

5.1.2 Proposed method

Formulation of the imperfect dataset problem

We define the imperfect dataset problem as sub-problems including the missing data, class imbalance, and missing label problems. To formulate these sub-problems, let us first define the set of data $\mathcal{D}$ relating to participants. $\mathcal{D}$ consists of MRI scans I, tabular features $\mathbf{t} \in \mathbb{R}^d$ $(d = 27)$, and amyloid positivities $y \in \{0, 1\}$. We denote the numbers of individuals in Aβ+ $(y = 1)$ and A$\beta-$ $(y = 0)$ groups as n_1 and n_0 respectively. $N_l = n_1 + n_0$ is the number of labeled participants, and N_u is the number of unlabeled participants. We now introduce two Boolean objects $\mathbf{m} \in \{0, 1\}^d$ and $o \in \{0, 1\}$ that represent the missingness of tabular features and labels. We also introduce y_g to represent the minority class, which is the Aβ+ group in this study. We can now divide $\mathcal{D}$ into labeled data (denoted as $\mathcal{D}_l = \{(\mathbf{t}^n, \mathbf{m}^n), I^n, (y^n, o^n = 1)\}_{n=1}^{N_l}$), and unlabeled data (denoted as $\mathcal{D}_u = \{(\mathbf{t}^n, \mathbf{m}^n), I^n, o^n = 0\}_{n=1}^{N_u}$).

Variables created from labeled data are marked with the subscript l, unlabeled data with the subscript u, and synthesized data with the subscript g. Two or more subscripts are combined to simplify the notation (e.g., $\mathbf{t}_l$ and $\mathbf{t}_u$ can be combined into $\mathbf{t}_{lu}$).

The deep generative model

We modified HexaGAN in Chapter 3 to predict early amyloid pathology from the imperfect dataset described above. Our model consists of seven components (Figure 5.1(b) and (c)). The auxiliary encoder E_h is a new addition to HexaGAN which maps high-dimensional image data to a low-dimensional embedded vector $\mathbf{h}$. The auxiliary encoder and the remaining six components, employed in the HexaGAN originally, play different roles in classifying Aβ+ and A$\beta-$ groups, as explained below:

- E_h is an encoder that receives an image and synthesizes an embedded vector $\mathbf{h}$.
- E_H is an encoder that receives table data and $\mathbf{h}$ to synthesize hidden vector $\mathbf{H}$.
- G_{MI} is a generator that receives $\mathbf{H}$, fills missing data, and synthesizes $\hat{\mathbf{h}}$ (for conditional generation).
- D_{MI} is a discriminator that distinguishes between real (non-missing) and fake (missing) elements of imputed data and embedding vector elements.
- G_{CG} is a generator that receives (minority) class labels and generates a hidden vector $\mathbf{H}_g$.
- D_{CG} is a discriminator that receives class labels and hidden vectors $\mathbf{H}$ to distinguish between real and fake hidden vectors.
- C is a classifier that predicts amyloid positivity by receiving imputed data $\hat{\mathbf{t}}$ and an embedded vector $\mathbf{h}$.

Our model receives 18 neuropsychological variables and 18 slices from the total 210 coronal slices via decimation as input. Among them, MRI slices I_l and I_u go through E_h to create embedded vectors $\mathbf{h}_l$ and $\mathbf{h}_u$, respectively (Figure 5.1(b)). In the case of tabular data $\mathbf{t}_l$ and $\mathbf{t}_u$, we replace missing elements with noise using a Hadamard product as follows:

$$\mathbf{h}_{lu} = E_h(I_{lu}) \tag{5.1}$$

$$\tilde{\mathbf{t}}_{lu} = \mathbf{m}_{lu} \circ \mathbf{t}_{lu} + (1 - \mathbf{m}_{lu}) \circ \mathbf{z}_{lu}, \tag{5.2}$$

where the operator $\circ$ indicates the Hadamard product of two vectors, and $\mathbf{z}_l$ and $\mathbf{z}_u \in [0, 1]^d$ are random noise sampled from $U(0, 1)$.

Missing data imputation

The components used for missing data imputation are E_h, E_H, G_{MI}, and D_{MI} (MI stands for 'missing imputation'). In this chapter, the subscript of all variables without a subscript is lu. To impute missing elements, E_H receives $\tilde{\mathbf{t}}$, $\mathbf{h}$, and $\mathbf{m}$ and creates the hidden vector $\mathbf{H} \in [-1, 1]^{d_H}$ (Figure 5.1(c)). Then G_{MI} receives $\mathbf{H}$ and creates $\bar{\mathbf{t}}$ and $\hat{\mathbf{h}} \in [0, 1]^{d_h}$. For tabular data, only the missing elements of $\mathbf{t}$ are replaced with $\bar{\mathbf{t}}$, and imputed tabular data is called $\hat{\mathbf{t}}$:

$$\mathbf{H} = E_H(\tilde{\mathbf{t}}, \mathbf{h}, \mathbf{m}) \tag{5.3}$$

$$(\bar{\mathbf{t}}, \hat{\mathbf{h}}) = G_{\mathrm{MI}}(\mathbf{H}) \tag{5.4}$$

$$\hat{\mathbf{t}} = \mathbf{m} \circ \mathbf{t} + (1 - \mathbf{m}) \circ \bar{\mathbf{t}}, \tag{5.5}$$

$\mathbf{h}$ and the synthesized $\hat{\mathbf{t}}$ are fed into D_{MI} with the class label. For labeled data, y is used as the class label, and for unlabeled data, the prediction from the classifier C (e.g., for binary class, $\lfloor C(\hat{\mathbf{t}}_u, \mathbf{h}_u) + 0.5 \rfloor$) is used and called y_u. Then, D_{MI} computes ($d + d_h + 1$) outputs, each of which is a value that measures whether elements of $\hat{\mathbf{t}}$, elements of $\mathbf{h}$, and labels are missing (fake) or non-missing (real). For missing data imputation, ($d + d_h$) elements of the output of D_{MI} (except the last element for a label) are used for loss functions. E_H and G_{MI} are learned via element-wise WGAN hinge loss as follows:

$$\begin{aligned} \mathcal{L}_{G_{\mathrm{MI}}}^{lu} &= -\mathbb{E}_{\hat{\mathbf{t}}|\mathbf{h},y,\mathbf{m}} \left[\frac{1}{\sum_{i=1}^{d+d_h} 1 - m_i^{th}} \sum_{i=1}^{d+d_h} (1 - m_i^{th}) \cdot D_{\mathrm{MI}}(\hat{\mathbf{t}}, \mathbf{h}, y)_i \right] && (5.6) \\ &= -\mathbb{E}_{\hat{\mathbf{t}}|\mathbf{h},y,\mathbf{m}} \left[\frac{1}{\sum_{i=1}^{d} 1 - m_i} \sum_{i=1}^{d} (1 - m_i) \cdot D_{\mathrm{MI}}(\hat{\mathbf{t}}, \mathbf{h}, y)_i \right], && (5.7) \end{aligned}$$

where $D_{\mathrm{MI}}(\cdot)_i$ is the i-th element of the output of D_{MI} and $\mathbf{m}^{th}$ denotes the missingness labels for the elements of $\hat{\mathbf{t}}$ and $\mathbf{h}$. If the element is missing, the label is marked as 0, and if it is non-missing, the label is 1. For labeled and unlabeled data, the missingness label for $\hat{\mathbf{t}}$ ($\mathbf{m}^t$) is $\mathbf{m}$, and the missingness label for $\mathbf{h}$ ($\mathbf{m}^h$) is $\mathbf{1}$ and $(\mathbf{1} - \mathbf{m}^h) = \mathbf{0}$, because there is no missing element in the image data. Therefore, $\mathcal{L}^{lu}_{G_{\mathrm{MI}}}$ uses only the first d elements for loss. D_{MI}, which plays a critical role for missing data imputation, uses the following element-wise adversarial loss:

$$\mathcal{L}^{lu}_{D_{\mathrm{MI}}} = -\,\mathbb{E}_{\hat{\mathbf{t}}|\mathbf{h},y,\mathbf{m}}\left[\frac{1}{\sum_{i=1}^{d+d_h} m_i^{th}} \sum_{i=1}^{d+d_h} m_i^{th} \cdot \min(0, -1 + D_{\mathrm{MI}}(\hat{\mathbf{t}}, \mathbf{h}, y)_i)\right] \tag{5.8}$$

$$-\,\mathbb{E}_{\hat{\mathbf{t}}|\mathbf{h},y,\mathbf{m}}\left[\frac{1}{\sum_{i=1}^{d+d_h} 1 - m_i^{th}} \sum_{i=1}^{d+d_h} (1 - m_i^{th}) \cdot \min(0, -1 - D_{\mathrm{MI}}(\hat{\mathbf{t}}, \mathbf{h}, y)_i)\right]. \tag{5.9}$$

In terms of a distance metric, Wasserstein distance is weaker than the Jensen-Shannon divergence used in a vanilla GAN [51], so it converges better with complex data distributions [8], and facilitates the imputation of missing values in the data. We expanded a projection layer [111] in an element-wise manner to inject the conditional information of y into D_{MI}.

We also used a reconstruction loss that improves the imputation process by learning from non-missing data as follows:

$$\mathcal{L}_{\mathrm{recon}} = \mathbb{E}_{\hat{\mathbf{t}},\hat{\mathbf{h}}|\mathbf{m},\mathbf{t},\mathbf{h}}\left[\frac{1}{d}\sum_{i=1}^{d} m_i \cdot (t_i - \hat{t}_i)^2 + \frac{1}{d_h}\sum_{i=1}^{d_h} m_i^h \cdot (h_i - \hat{h}_i)^2\right] \tag{5.10}$$

$$= \mathbb{E}_{\hat{\mathbf{t}},\hat{\mathbf{h}}|\mathbf{m},\mathbf{t},\mathbf{h}}\left[\frac{1}{d}\sum_{i=1}^{d} m_i \cdot (t_i - \hat{t}_i)^2 + \frac{1}{d_h}\sum_{i=1}^{d_h} (h_i - \hat{h}_i)^2\right], \tag{5.11}$$

where $\mathbf{m}^h = \mathbf{1}$, so it can be omitted. Therefore, E_H is trained to optimize the following objective:

$$\min_{E_H} \mathcal{L}^{lu}_{G_{\mathrm{MI}}} + \lambda_1 \mathcal{L}_{\mathrm{recon}}, \tag{5.12}$$

where λ_1 is a hyperparameter that adjusts the ratio between losses: a value of 10 was used for this experiment. (Loss functions for G_{MI} are also related to class conditional generation; therefore, they will be described later.)

Class conditional generation

The class imbalance problem is addressed by G_{CG} and D_{CG}, which synthesize hidden vectors conditioned on the minority class label y_g, and by G_{MI}, which oversamples the minority class by imputing all features. G_{CG} (CG stands for 'conditional generation') receives the noise vector $\mathbf{z}_g \in [0,1]^d$ sampled from $U(0,1)$ and the minority class label y_g, and creates the hidden vector $\mathbf{H}_g$. Then G_{MI} receives $\mathbf{H}_g$ and synthesizes the oversampled data $\hat{\mathbf{t}}_g$ and $\hat{\mathbf{h}}_g$:

$$\mathbf{H}_g = G_{\mathrm{CG}}(\mathbf{z}_g, y_g) \tag{5.13}$$

$$(\hat{\mathbf{t}}_g, \hat{\mathbf{h}}_g) = G_{\mathrm{MI}}(\mathbf{H}_g). \tag{5.14}$$

Here, $\hat{\mathbf{t}}_g$ is computed on the basis that all elements as missing, and $\hat{\mathbf{h}}_g$ is not obtained from a real image. Therefore, the missingness labels $\mathbf{m}_g^t$ and $\mathbf{m}_g^h$ are zero vectors $\mathbf{0}$ in the d and d_h dimensions, respectively.

D_{CG} is trained to minimize WGAN hinge loss, considering $\mathbf{H}_l$ as real data and $\mathbf{H}_g$ as fake data, as follows:

$$\mathcal{L}_{D_{\mathrm{CG}}} = \mathbb{E}_{\mathbf{H}_l|y}\left[\max(0, 1 - D_{\mathrm{CG}}(\mathbf{H}_l, y))\right] + \mathbb{E}_{\mathbf{H}_g|y_g}\left[\max(0, 1 + D_{\mathrm{CG}}(\mathbf{H}_g, y_g))\right] \tag{5.15}$$

$$\min_{D_{\mathrm{CG}}} \mathcal{L}_{D_{\mathrm{CG}}}. \tag{5.16}$$

To inject the conditional information of y into D_{CG}, we used a projection layer [111].

The goal of G_{CG} can be subdivided into three. The first is to synthesize a realistic $\mathbf{H}_g$, which can be learned through an adversarial loss so that D_{CG} can be fooled by G_{CG}. The second is to synthesize realistic $\hat{\mathbf{t}}_g$ and $\hat{\mathbf{h}}_g$, which can be learned through element-wise adversarial loss to deceive D_{MI}. Third, since oversampled data should be well-conditioned on the minority class, a prediction is obtained by feeding $\hat{\mathbf{t}}_g$ and $\hat{\mathbf{h}}$ to C, and then trying to minimize the cross-entropy between the prediction and y_g.

G_{CG} is trained to minimize the following three loss functions:

$$\mathcal{L}_{G_{\text{CG}}} = -\mathbb{E}_{\mathbf{H}_g|y_g}\left[D_{\text{CG}}(\mathbf{H}_g, y_g)\right] \tag{5.17}$$

$$\mathcal{L}^g_{G_{\text{MI}}} = -\mathbb{E}_{\hat{\mathbf{t}}_g,\hat{\mathbf{h}}_g|y_g}\left[\frac{1}{d+d_h}\sum_{i=1}^{d+d_h} D_{\text{MI}}(\hat{\mathbf{t}}_g, \hat{\mathbf{h}}_g, y_g)_i\right] \tag{5.18}$$

$$\mathcal{L}^g_C = \mathbb{E}_{\hat{\mathbf{t}}_g,\hat{\mathbf{h}}_g|y_g}\left[-y_g \cdot \log C(\hat{\mathbf{t}}_g, \hat{\mathbf{h}}_g)\right] \tag{5.19}$$

$$\min_{G_{\text{CG}}} \mathcal{L}_{G_{\text{CG}}} + \lambda_2 \mathcal{L}^g_{G_{\text{MI}}} + \lambda_3 \mathcal{L}^g_C, \tag{5.20}$$

where λ_2 and λ_3 are hyperparameters used as coefficients for $\mathcal{L}^g_{G_{\text{MI}}}$ and $\mathcal{L}^g_C$ respectively. In experiments, we used $\lambda_2 = 1$ and $\lambda_3 = 0.01$. Since missingness labels for $\hat{\mathbf{t}}_g$ and $\hat{\mathbf{h}}_g$ are 0, $\mathcal{L}^g_{G_{\text{MI}}}$ can be organized as above. In addition, as a critic for the synthesized data, D_{MI} is also trained through adversarial loss, as follows:

$$\mathcal{L}^g_{D_{\text{MI}}} = -\mathbb{E}_{\hat{\mathbf{t}}|\mathbf{h},y,\mathbf{m}}\left[\frac{1}{d+d_h}\sum_{i=1}^{d+d_h} \min(0, -1 - D_{\text{MI}}(\hat{\mathbf{t}}, \mathbf{h}, y)_i)\right]. \tag{5.21}$$

We now have a missingness label for all labeled, unlabeled, and synthesized data. The adversarial loss for labeled and unlabeled data is $\mathcal{L}_{G^{lu}_{\text{MI}}}$, while that for synthesized data is $\mathcal{L}^g_{G_{\text{MI}}}$. In addition, we can use a reconstruction loss for non-missing data to learn G_{MI} as follows:

$$\min_{G_{\text{MI}}} \mathcal{L}^{lu}_{G_{\text{MI}}} + \mathcal{L}^g_{G_{\text{MI}}} + \lambda_1 \mathcal{L}_{\text{recon}}, \tag{5.22}$$

where λ_1, the coefficient for the reconstruction loss, was set to 10, the same value used to update E.

Classification and semi-supervised learning

The missing label problem is addressed by semi-supervised learning of two components; the classifier C acts as a generator and p_y of D_{MI} acts as a discriminator for the pseudo-labeling of unlabeled data in order to improve classification performance. When a mini-batch of data enters the model, we set the number of y_g as the difference between the amount of majority class data and that of majority class data in the mini-batch. Thus, the class labels of $\mathbf{s}_l$ and $\mathbf{s}_g$ together are now balanced and used to train the classifier C, where input features $\mathbf{s} = (\mathbf{h}, \hat{\mathbf{t}})$. Then, the mini-batch and oversampled data are used for calculating the cross-entropy loss as follows:

$$\mathcal{L}_C^{lg} = \mathbb{E}_{\hat{\mathbf{t}}_l|\mathbf{h}_l,y_l}\left[-y_l \cdot \log C(\mathbf{s}_l)\right] + \mathbb{E}_{\mathbf{s}_g|y_g}\left[-y_g \cdot \log C(\mathbf{s}_g)\right]. \tag{5.23}$$

In addition, C can synthesize a pseudo-label y_u for $\hat{\mathbf{t}}_u$ and $\mathbf{h}_u$. The synthesized pseudo-label can be assessed by D_{MI}, as follows:

$$\mathcal{L}_C^{y} = -\mathbb{E}_{y_u|\hat{\mathbf{t}}_u,\mathbf{h}_u \sim p_C}\left[D_{\text{MI}}(\hat{\mathbf{t}}_u, \mathbf{h}_u, \mathbf{y}_u)_{d+d_h+1}\right] \tag{5.24}$$

$$\mathcal{L}_{D_{\text{MI}}}^{y} = \mathbb{E}_{y_u|\hat{\mathbf{t}}_u,\mathbf{h}_u \sim p_C}\left[D_{\text{MI}}(\hat{\mathbf{t}}_u, \mathbf{h}_u, y_u)_{d+d_h+1}\right] - \mathbb{E}_{y_l|\hat{\mathbf{t}}_l,\mathbf{h}_l}\left[D_{\text{MI}}(\hat{\mathbf{t}}_l, \mathbf{h}_l, y)_{d+d_h+1}\right], \tag{5.25}$$

where p_C is the distribution of the classifier. It has been shown that such adversarial learning helps supervised learning by minimizing the ODM cost [70, 148]. Finally, C and D_{MI} are trained with the following objectives:

$$\min_C \mathcal{L}_C^{lg} + \lambda_4 \mathcal{L}_C^{y} \tag{5.26}$$

$$\min_{D_{\text{MI}}} \mathcal{L}_{D_{\text{MI}}}^{lu} + \mathcal{L}_{D_{\text{MI}}}^{g} + \lambda_5 \mathcal{L}_{D_{\text{MI}}}^{y}. \tag{5.27}$$

We set λ_4 and λ_5 to 0.005 for our experiments.

The encoder E_h should synthesize the embedded vector $\mathbf{h}$ that serves to make the classifier C achieve a high classification performance. In addition, E_h should also help E_H and G_{MI} synthesize the over-sampled embedded vector $\hat{\mathbf{h}}$ whose distribution resembles that of $\mathbf{h}$. Thus, we trained E_h using the following loss function $\mathcal{L}_{E_h}$ which is a linear combination of a reconstruction loss function $\mathcal{L}^h_{\text{recon}}$ and a cross-entropy loss function $\mathcal{L}^l_C$:

$$\mathcal{L}^h_{\text{recon}} = \|\mathbf{h} - \hat{\mathbf{h}}\|_2^2 \tag{5.28}$$

$$\mathcal{L}^l_C = \left[-y \cdot \log C(\mathbf{s}_l)\right]. \tag{5.29}$$

Therefore, E_h is trained to optimize the following objective:

$$\min_{E_h} \mathcal{L}^h_{\text{recon}} + \mathcal{L}^l_C. \tag{5.30}$$

Our method has additional methodological contributions to HexaGAN. The original HexaGAN uses a zero-centered gradient penalty [104] to stabilize adversarial learning. However, keeping the gradient near zero makes training slow, and this is further reduced by the need to calculate penalties for high-dimensional data on an element-by-element basis. We apply spectral normalization [113] to the weights of all the layers in our network, which achieves stable convergence much more quickly. To improve the generalization of the encoder E_h, we employed DenseNet pretrained on the ImageNet dataset [67, 121]. Representations learned from a large-scale dataset help to improve generalization performance as a rule of thumb even for a specific medical application with data having different input dimensions and the number of channels [142]. The network architecture of each component and training details are described in Table 5.7 and Section 5.1.6.

Model comparison

When training our model with MRI scans alone, E_H and G_{MI} are not required, and G_{CG} synthesizes embedded vectors $\mathbf{h}$ instead of hidden vectors $\mathbf{H}$. When training our model with only tabular data, E_h is not used. We used various machine learning methods for comparison. For imperfect dataset problems, we constructed several benchmark models consisting of the same classifier as C of our model. To solve the imperfect dataset problem for the benchmark models, we used column-wise deletion, mean imputation, k-nearest neighbor (kNN) imputation [157] and multiple imputation by chained equation (MICE) [21] to deal with the missing data problem; kNN and label propagation [179] to deal with the missing label problem; and synthetic minority oversampling technique (SMOTE) [24], adaptive synthetic (ADASYN) method [64], cost sensitive loss [144] and class rectification loss [38] to deal with the class imbalance problem. Since all labeled data have one or more missing values, it is impossible to deal with missing values via a list-wise deletion when both labeled and unlabeled data were used. We computed the area under the receiver operating characteristic (AUROC) curve estimated by 5-fold cross validation implemented in Scikit-learn [123] to evaluate generalized classification performance. We also performed feature ablation trials on a partition of the feature set consisting of MRI scan and tabular data to check our assumption that combining these types of data improves performance.

We evaluated the effect of imputation on the classification performance of our model as part of our exploratory process of determining where high accuracy came from. We randomly removed between 10% and 90% of observed tabular feature values and imputed these missing values using our model and the aforementioned imputation techniques. This process was repeated 100 times, each starting at a different random state for each missing rate. We then computed the 5-fold average root mean square errors (RMSE) between the imputed and original values.

Analysis by t-stochastic neighbor embedding

We inspected the changes which occurred to the spaces of the hidden features ($\mathbf{H}$) and input features ($\mathbf{s}$) as the model was trained. For this analysis, we trained a randomly initialized model with all the data. At each epoch, we visualized the observed hidden features $\mathbf{H}_l$ synthesized by E_h and E_H; and the synthetic hidden features $\mathbf{H}_g$

synthesized by G_{CG} by reducing their dimensionality using t-stochastic neighbor embedding (t-SNE) [102]. We visualized the labeled input features $\mathbf{s}_l = (\hat{\mathbf{t}}_l, \mathbf{h}_l)$, and the synthesized input features $\mathbf{s}_g = (\hat{\mathbf{t}}_g, \hat{\mathbf{h}}_g)$ in a similar way.

Imputing missing features

We tested the ability of our model to cope with features that are missing entirely. We first trained our model with all tabular and image features. We then measured the 5-fold average of test AUROCs using the fully trained model. Then we greedily removed tabular features of test data, starting with the feature that produces the largest reduction in the AUROC.

Discriminative regions and variables

Feature attributions allow us to quantify how discriminative regions and variables are. We first chose the model that produced the best predictions out of the five sets of models learned during stratified 5-fold cross validation. We then trained the model from these models using the whole of each dataset until training AUROC reached 1 from the chosen model. We then computed *feature attributions* A^n for n-th participant using the integrated gradient (IG) algorithm [145]:

$$A^n = \frac{\mathrm{IG}_f\left(X^n, X_{\min}\right) + \mathrm{IG}_f\left(X^n, X_{\max}\right)}{2}, \tag{5.31}$$

where IG_f is an integrated gradient functional depending on a deep learning model f, $X^n = (\mathbf{t}^n, I^n)$ is an observed features which consists of tabular features and an MRI scan, and $X_{\min}$ and $X_{\max}$ are features that have the same size as X^n and have the values all 0 and 1, respectively. $\mathrm{IG}_f(X^n, X_{\min})$ was calculated by summing the gradients of the model output with respect to a sequence of $k+1$ progressively interpolated features between $X_{\min}$ and X, as follows:

$$\mathrm{IG}_f(X^n, X_{\min}) = (X^n, X_{\min}) \times \int_0^1 \nabla f\left(\alpha \times X^n + (1-\alpha) \times X_{\min}\right) d\alpha, \quad (5.32)$$

$$\approx (X^n, X_{\min}) \times \sum_{i=0}^{k} \nabla f\left(\frac{i}{k} \times X^n + \frac{k-i}{k} \times X_{\min}\right). \quad (5.33)$$

$\mathrm{IG}_f(X^n, X_{\max})$ was calculated in similar manner.

For tabular data $\mathbf{t}$, we define the *importance* of the i-th tabular feature t_i as $\mathcal{I}_{t_i}^n = (A_{t_i}^n)^2$, where $A_{t_i}^n$ is the attribution value of tabular feature t_i in the attribution map A^n. The importance value represents the amount of influence for predicting early amyloid pathology regardless of its sign.

For MRI scans I, attributions are computed at region-of-interest (ROI) level, not at voxel level, according to XRAI framework [77]. Instead of Felzenszwalb's graph-based method used in the original XRAI paper for determining parcels, we used an atlas-tased segmentation technique with the HM atlas that structurally divides a brain image, including cerebral gray/white matters, ventricles, cerebellum, and brainstem into pre-defined regions. Similar with tabular data, we defined the *importance* of i-th brain region r_i, as follows:

$$\mathcal{I}_{r_i}^n = \left(\frac{1}{n_{r_i}} \sum_{v \in r_i} A_{I_v}\right)^2, \quad (5.34)$$

where A_{I_v} is the attribution value of voxel v in MRI scan I, and n_{r_i} is the number of voxels of a brain region r_i. Since we are interested in brain regions that affect the early amyloid pathology as tabular features, we focused on the amount of influence for each region regardless of its direction that represents whether the region should be bright or dark to be classified into the correct class. We consider population-level attribution as the square root of the average importance value over labeled participants.

Finally, we defined discriminative regions on variables as ROIs or tabular features with statistically significant attributions. Statistical significance is determined based on Bonferroni-corrected 95% confidence intervals for the population-level attribution computed by the studentized bootstrap procedure with 10,000 bootstrap resamples.

5.1.3 Experimental setup

Participants

The discovery dataset was obtained from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database[1]. The discovery dataset was used to construct the model to address the imperfect dataset problem, and to verify that our model can robustly predict amyloid positivity. It includes 539 cognitively normal (CN) participants enrolled in the ADNI-1, ADNI-GO, ADNI-2, and ADNI-3 cohorts, who had T1-weighted magnetic resonance imaging (MRI) scans from a screening visit. The details of the ADNI design, participant recruitment, and diagnostic criteria are published on the ADNI website[2].

The practice dataset was obtained from the Samsung Medical Center (SMC). The practice dataset consists of features and diagnostic criteria different from the discovery dataset, and was used to verify that our model trained with data from a different cohort (in our study, the discovery dataset) can make good predictions on data from one hospital. It includes 343 CN participants who had T1-weighted MRI scans. All participants underwent a detailed clinical interview and a neurologic examination [28]. The study protocol was approved by the Institutional Review Board of the SMC.

MRI acquisition

To construct the discovery dataset, we downloaded T1-weighted MRI scans for 539 participants from ADNI's database. Specifically, we used 1.5 T non-accelerated magnetization-prepared rapid acquisition gradient echo (MP-RAGE) scans for the ADNI-1 cohort, 3 T non-accelerated MP-RAGE or inversion recovery spoiled gradient echo (IR-SPGR) scans for the ADNI-2 and ADNI-GO cohorts, and 3 T non-accelerated MP-RAGE

[1] http://www.adni.loni.usc.edu

[2] http://www.adni.loni.usc.edu/methods/

scans for the ADNI-3 cohort. Details of the MRI protocols employed are available on the ADNI website.

For T1-weighted MRI scans of the practice dataset, we used 3 T turbo field echo images acquired at SMC. Detail information about acquisition protocols is described in the previous study [28].

Image preprocessing

T1-weighted MRI scans acquired from 539 participants were preprocessed according to the standard procedures of image preprocessing with FreeSurfer v.6.0.0[3]. These include nonlinear registration to Talairach space, intensity correction, skull stripping, and anatomical segmentation of entire brain regions (including the cerebral gray matter, white matter, and cerebellum) according to the Hammersmith (HM) atlas [52]. The intracranial volume (ICV) of each subject was computed for further analysis.

Originally, all the MRI scans that were non-linearly transformed to Talairach space had a resolution of 256×256×256. Due to the limitation of our graphical processing unit (GPU) memory, however, we performed the two following steps. First, we reduced the field-of-view (FOV) of our MRI scan volumes to 168×190×210, excluding the non-brain background. We then extracted every 10th slice, from the 20th to the 190th slice. Thus, the intensities of MRI scans had a resolution of 168×190×18. MRI scans were normalized to the range of 0-1.

After preprocessing, one participant was excluded from the ADNI dataset due to an image preprocessing failure, therefore the final number of participants used in this study is 538.

Amyloid positivity determination

For the discovery dataset, global amyloid positivity was determined from positron emission tomography (PET) scans, which consisted of four 5 min frames, acquired

[3] http://surfer.nmr.mgh.harvard.edu

50–70 min after an injection of 370 ± 37 MBq of [^{18}F]-AV45. Amyloid positivity is defined as a cortical AV45 standardized uptake value ratio (SUVR) > 1.11 [90]. AV45 SUVRs were average values of frontal, anterior cingulate, precuneus, and parietal cortex relative to the cerebellum, which were extracted from the ADNIMERGE file. Further details of PET imaging protocols are published on the ADNI website and elsewhere [90].

For the practice dataset, global amyloid positivity was determined by the scoring system as described in the previous study [44], where three medical experts visually assessed [^{18}F]-Florbetaben (FBB) PET scans [61] acquired using PET/CT scanner at SMC.

Tabular data

The tabular data consists of a total of 27 demographic, genetic, and clinical variables relating to individuals. The demographic variables are age, sex, years of education, intracranial volume, and handness. The genetic variables are the number of APOE $\varepsilon 4$ alleles. The 21 clinical variables measure cognitive function and instrumental activities of daily living (IADL).

Three of the clinical variables measure AD-related global cognitive impairment: a mini-mental state examination (MMSE) score; a clinical dementia rating sum of boxes (CDR-SB); and a score for the Alzheimer's disease assessment scale, 13-item version (ADAS13). In addition, 18 variables were extracted from the ADNI neuropsychological battery considering redundancy in cognitive domain [12], which are listed in Table 5.2. From the results of a Rey auditory verbal learning test (AVLT), we created learning and memory variables by using the 'Immediate recall', 'Delayed recall', and 'Recognition' scores, and by computing 'Learning', 'Intrusion error (IntErr)', 'Proactive interference (ProINTFC)', and 'Retroactive interference (RetroINTFC)' scores [154]. Finally, two variables were created from measurements of instrumental activities of daily living (IADL): scores from a functional activity questionnaire (FAQ) and for everyday cognition assessed by the patient (ECogPt). Where more than one set of scores was available for the period within 90 days of a participant's MRI scan, we chose the set with the lowest proportion of missing entries.

All 27 variables in the tabular data were normalized to the range 0-1.

5.1.4 Experimental results

Dataset information

Table 5.1 shows the demographic, genetic, and clinical characteristics of the participants in our discovery and practice datasets. In the discovery datasets, APOE $\epsilon 4$, age, and sex differed significantly between the amyloid positive (Aβ+) and negative (A$\beta-$) groups. In the practice dataset, APOE $\epsilon 4$ and age were significantly different between the two groups, as in the discovery dataset.

Three aspects of the imperfect dataset problem are relevant to this study. Firstly, all the records in the discovery dataset lack at least one of the 18 neuropsychological variables, which include measures of cognitive functioning and instrumental activities of daily living (IADL). These absences are concentrated in records without a diagnosis (Table 5.2). In the practice dataset, not only does the tabular data have missing values, but no neuropsychological variables were observed. Secondly, 260 records in the discovery dataset and 187 in the practice dataset correspond to participants who did not have amyloid PET scans, so that amyloid positivity is unknown in almost half of the records. These records were used as unlabeled data in the later process. Thirdly, the ratios between the sizes of the Aβ+ and A$\beta-$ groups are about 1:2 and 1:5 in the discovery and practice datasets respectively.

Improved classification performance with imperfect datasets

Our model achieved an average AUROC for 5-fold cross validation of 0.8609, which was 17.2% higher than the best of the other DL models such as multilayer perceptrons

Table 5.1: Demographic and clinical characteristics of participants.

	Discovery dataset				Practice dataset			
Characteristics	**Total**	**Aβ+**	**Aβ−**	**p values**	**Total**	**Aβ+**	**Aβ−**	**p values**
N	538	93	186		343	28	128	
Age, years[†]	74.21 (5.84)	74.68 (5.87)	72.32 (5.98)	0.0020*	66.76 (12.04)	71.00 (6.54)	64.77 (15.85)	0.0012*
Sex (female)[‡], no.	128	30	98	0.0019*	217	18	78	0.9081
Education, years[†]	16.38 (2.68)	16.15 (2.70)	16.91 (2.43)	0.0178	11.52 (4.69)	11.46 (4.39)	11.43 (4.83)	0.9728
ICV, ℓ[†]	1.50 (0.16)	1.47 (0.17)	1.49 (0.16)	0.6111	1.52 (0.17)	1.58 (0.18)	1.53 (0.16)	0.2201
Handness (LH), no.[‡]	28	10	18	0.9438	N/A[#]	N/A	N/A	N/A
APOE ε4 (0/1/2), no.[‖]	369/137/12	50/39/4	147/36/3	< 0.001*	253/73/9	8/16/4	103/20/1	< 0.001*
MMSE[†]	29.06 (1.14)	29.03 (0.96)	29.00 (1.32)	0.8168	28.07 (2.04)	27.68 (1.56)	28.22 (1.53)	0.0960
CDR-SB[†]	0.040 (0.139)	0.065 (0.184)	0.040 (0.137)	0.2632	0.677 (0.694)	0.893 (0.600)	0.659 (0.470)	0.0276
ADAS13[†]	9.23 (4.33)	9.65 (4.42)	8.68 (4.37)	0.0836	N/A	N/A	N/A	N/A

N, Sample size; no. Number; ICV, Intracranial volume; MMSE, Mini-mental state examination; CDR-SB, Clinical dementia rating, sum of boxes; ADAS13, Alzheimer's disease assessment scale, 13-item version.

[†] Data are given as the mean and standard deviation; p values were calculated by two-sided Student's or Welch's two-sample t-tests.

[‡] For dichotomous data, p values were calculated by Yates-corrected Chi-square tests.

[‖] Data are given as frequencies of the numbers of alleles; p value was calculated by a two-sided Fisher's exact test.

[#] Statistics are not available because features are not observed in the dataset.

[*] p <0.05. Significance adjusted after Bonferroni correction.

(MLP) and convolutional neural networks (CNN) (Figure 5.2(a)). At optimal cut-offs determined by the Youden index [172], our model showed an average accuracy of 0.8244, an average sensitivity of 0.8415, and an average specificity of 0.8178. Performance on each fold of our model and comparative results can be found in Tables 5.3 and 5.4, respectively.

Synergistic effects of image and tabular data

When trained with just one of these types of data, our model produced more accurate classifications than other DL models (Figure 5.2(b) and (c)); and these classifications improved when our model was trained with both types of data. This contrasts with our observations of discriminative DL models, which do not perform much differently when MRI scans are added to tabular data. Further details of these comparisons are

Table 5.2: Neuropsychological variables fed into our deep generative model, with rates of missing data.

		Missing rates in discovery dataset (%)			
Domain	**Measure**	**Aβ+**	**Aβ−**	**Label missing**	**Total**
Language	BNTTOTAL	0	0	10.81	5.20
	CATANIMSC	0	0	10.04	4.83
Learning & memory	AVLT-immediate	0	0.54	11.58	5.76
	AVLT-delayed	0	0	10.42	5.02
	AVLT-recog	0	0	10.42	5.02
	Learning	0	0	10.81	5.20
	IntErr	0	0.54	11.58	5.75
	ProINTFC	0	0	10.81	5.20
	RetroINTFC	0	0.54	11.58	5.76
Attention	TRAASCOR	0	0	10.04	4.83
	DIGITSCOR	100	100	14.29	58.74
Executive	TRABSCOR	0	0.54	10.04	5.02
	DSPANFOR	100	100	14.29	58.74
	DSPANBAC	100	100	14.29	58.74
Perceptual-motor	CLOCKSCOR	0	0	10.04	4.83
	COPYSCOR	0	0	10.42	5.02
IADL	FAQ	0	0	10.81	5.20
	ECogPT	1.08	0	95.75	46.28

BNTTOTAL, Boston naming test, total score; CATANIMSC, Category fluency, animals; AVLT, Rey auditory verbal learning test; IntErr, intrusion error; ProINTFC, proactive interference; ReteroINTFC, retero interference [154]; TRAASCOR/TRABSCOR, trail making test, Part A/Part B; DIGITSCOR/DSPANFOR/DSPANBAC, digit span, digit symbol/forward#:total correct/backward#:total correct; CLOCKSCOR, clock drawing, total; COPYSCOR, clock copy, total; FAQ, functional activity questionnaire; ECogPT, everyday cognition, participants.

presented in Tables 5.4, 5.5, and 5.6.

Effectiveness of feature extraction

To evaluate the effectiveness of the encoder's feature extraction, we conducted comparative experiments between pretrained embeddings and our method on the discovery dataset. Specifically, we pretrained networks with the same architecture as our encoders (E_H and E_h) using either the reconstruction loss, the cross-entropy loss, or both. To mitigate the potential negative impact of the imperfect dataset problem during pretraining, we applied mean imputation and ADASYN, which were found to be the most effective

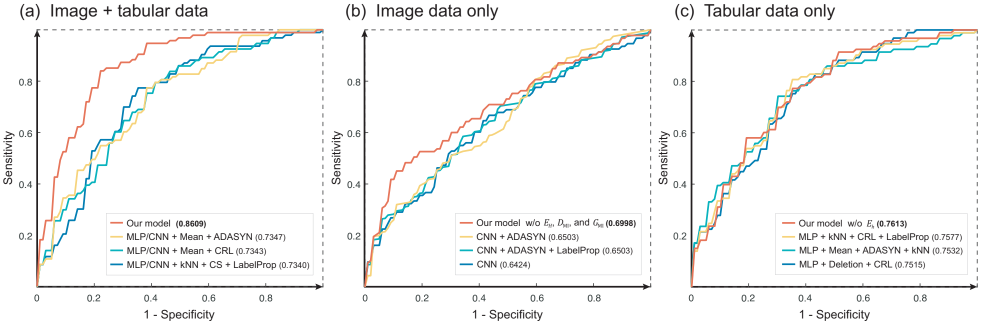


Figure 5.2: Classification Performance of our model. Average area under the receiver operating characteristic (AUROC) curves over 5-folds for our model and comparative classifiers, trained with both MRI scans and tabular data (a), with only MRI scans (b), and with only tabular data (c).

Table 5.4: Classification Performance with MRI Scans and Tabular Data.

Architecture	Missing data	Class imbalance	Missing label	Mean AUROC
MLP + CNN	Column-wise deletion	-	-	0.7043
		SMOTE	-	0.7111
		ADASYN	-	0.7231
		Cost-sensitive	-	0.7032
		Class rectification loss	-	0.7066
	Mean imputation	-	-	0.7099
		SMOTE	-	0.7104
		ADASYN	-	**0.7347**
		Cost-sensitive	-	0.7143
		Class rectification loss	-	**0.7343**
		ADASYN	k-nearest neighbors	0.6889
		ADASYN	Label propagation	0.6761
	k-nearest neighbors	-	-	0.7141
		SMOTE	-	0.7110
		ADASYN	-	0.7121
		Cost-sensitive	-	0.7280
		Class rectification loss	-	0.7221
		Cost-sensitive	k-nearest neighbors	0.7044
		Cost-sensitive	Label propagation	**0.7340**
	MICE	-	-	0.7180
		SMOTE	-	0.7182
		ADASYN	-	0.7100
		Cost-sensitive	-	0.7004
		Class rectification loss	-	0.7003
		SMOTE	k-nearest neighbors	0.6818
		SMOTE	Label propagation	0.6840
Our model	**HexaGAN**	**HexaGAN**	**HexaGAN**	**0.8609**

Table 5.5: Classification Performance with Tabular Data.

Architecture	Missing data	Class imbalance	Missing label	Mean AUROC
MLP	Column-wise deletion	-	-	0.7481
		SMOTE	-	0.7511
		ADASYN	-	0.7496
		Cost-sensitive	-	0.7492
		Class rectification loss	-	**0.7515**
	Mean imputation	-	-	0.7373
		SMOTE	-	0.7335
		ADASYN	-	0.7409
		Cost-sensitive	-	0.7311
		Class rectification loss	-	0.7339
		ADASYN	k-nearest neighbors	**0.7532**
		ADASYN	Label propagation	0.7445
	k-nearest neighbors	-	-	0.7409
		SMOTE	-	0.7402
		ADASYN	-	0.7419
		Cost-sensitive	-	0.7411
		Class rectification loss	-	0.7423
		Class rectification loss	k-nearest neighbors	0.7458
		Class rectification loss	Label propagation	**0.7577**
	MICE	-	-	0.7424
		SMOTE	-	0.7377
		ADASYN	-	0.7380
		Cost-sensitive	-	0.7345
		Class rectification loss	-	0.7392
		-	k-nearest neighbors	0.7441
		-	Label propagation	0.7503
Our model w/o E_h	**HexaGAN**	**HexaGAN**	**HexaGAN**	**0.7613**

Table 5.6: Classification Performance with MRI Scans.

Architecture	Class imbalance	Missing label	Mean AUROC
CNN	-	-	**0.6424**
	SMOTE	-	0.6158
	ADASYN	-	**0.6503**
	Cost-sensitive	-	0.6021
	Class rectification loss	-	0.6217
	ADASYN	k-nearest neighbors	0.6229
	ADASYN	Label propagation	**0.6503**
Our model w/o E_H, D_{MI}, and G_{MI}	**HexaGAN**	**HexaGAN**	**0.6998**

Table 5.3: Classification performance of our model over five folds.

Dataset	Fold	AUROC	Cut-off[*]	Sensitivity[†]	Specificity[‡]	Accuracy[#]
Discovery dataset	Fold 1	0.8757	0.0101	1.0000	0.7105	0.8036
	Fold 2	0.8165	0.1168	0.6842	0.8919	0.8214
	Fold 3	**0.8834**	**0.3193**	**0.7895**	**0.8649**	**0.8393**
	Fold 4	0.8720	0.0039	0.7895	0.8378	0.8214
	Fold 5	0.8574	0.0001	0.9444	0.7838	0.8364
Practice dataset	Fold 1	0.8333	0.7339	0.8333	0.8077	0.8125
	Fold 2	0.9000	0.0048	1.0000	0.7692	0.8065
	Fold 3	0.9385	0.0035	1.0000	0.8462	0.8710
	Fold 4	**0.9667**	**0.6872**	**1.0000**	**0.9200**	**0.9355**
	Fold 5	0.9333	0.0002	1.0000	0.8000	0.8387

AUROC, area under the receiver operating characteristic curve.

[*] Optimal cut-offs were determined by the Youden index.

[†] Sensitivity = (number of individuals correctly identified as Aβ+) / (total number of Aβ+ individuals).

[‡] Specificity = (number of individuals correctly identified as A$\beta-$) / (total number of A$\beta-$ individuals).

[#] Accuracy = (number of correctly identified individuals) / (total number of individuals).

BOLD: Folds where the highest AUROC was achieved.

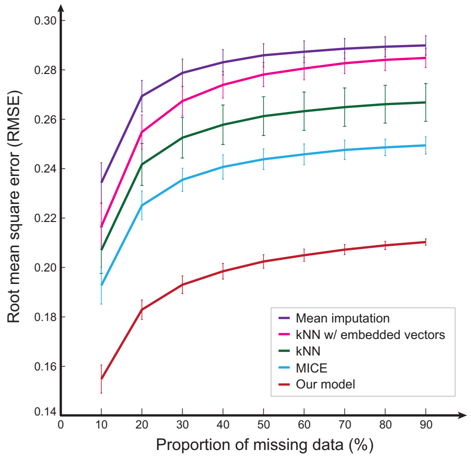
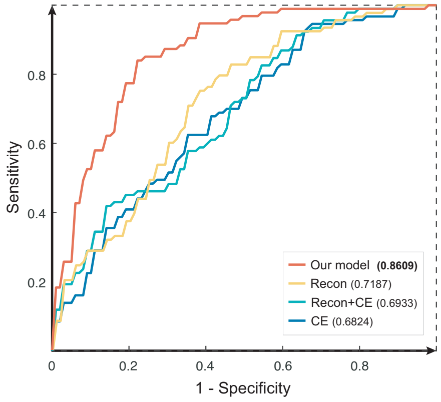


Figure 5.3: Performance comparison on the imputation of missing data. Performances of our model and comparative methods are measured using root mean square error (RMSE) values between the imputed values and the original values computed from 100 sets of test data. We removed 10% to 90% of features at random from test data. The error bar represents the standard deviation of RMSE values.



Method	AUROC	F1-score	PRAUC
Ours	**0.8609**	**0.7596**	**0.6421**
Recon	0.7187	0.6095	0.5869
Recon+CE	0.6933	0.5751	0.5713
CE	0.6824	0.5949	0.4593

Figure 5.4: Performance comparison on the feature extraction. Recon and CE denote the reconstruction loss and cross-entropy loss for pretraining, respectively. Average area under the receiver operating characteristic (AUROC) curves over 5-folds for our method and pretrained encoders (left). Performance comparison with AUROC, F1-score, and the area under the precision-recall curve (PRAUC) (right).

in Section 5.1.4. We then plugged in the pretrained networks, whose weights were frozen, instead of the encoders, and trained the remaining components of HexaGAN.

Figure 5.4 illustrates the results of the performance comparison. Our method outperformed pretrained embeddings across all performance metrics. We attribute this improvement to the fact that our method interacts with other components during training, particularly through the adversarial loss with the discriminator. Additionally, the use of the cross-entropy loss calculated on better-imputed data contributed to the enhanced performance.

Imputation of missing values

For our model, the average RMSE value increased from 0.16 to 0.21 as the proportion of missing data increased from 10% to 90%, suggesting our model was better at imputation even at 90% missing. For all proportions of missing data, over 100 trials, our model showed the lowest average RMSE values, followed by multiple imputation by chained equations (MICE), k-nearest neighbor (kNN) and mean imputation, in that order (Figure 5.3). In addition, our model showed the lowest standard deviation of RMSE values over

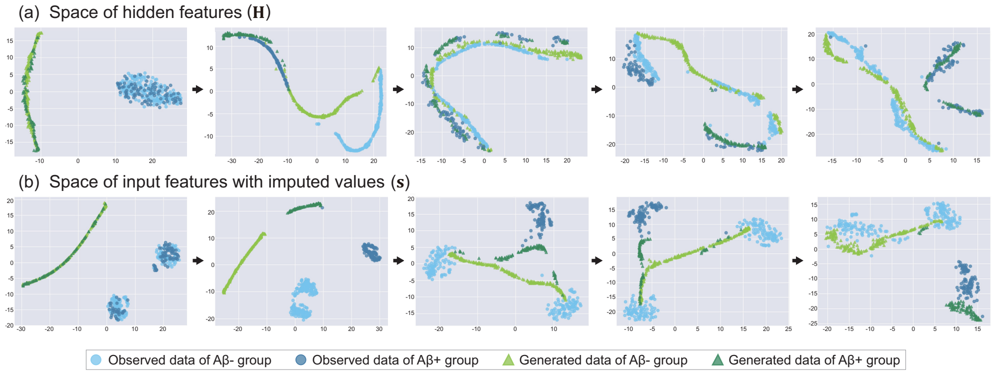


Figure 5.5: Assessment of the class conditional generation performance of our model. We use a t-stochastic neighbor embedding (t-SNE) analysis. Successive charts, from left to right, show the displacement of hidden features (a) and input features with imputed values (b) as the model is trained.

repeated trials at all missing rates compared to the other models. Taken this together, we verify that our model achieved the best classification performance by imputing missing values with more accurate and robust values.

We also found that, with all the methods, the RMSE increases rapidly as the proportion of missing data increases, and then becomes saturated (Figure 5.3). This seems to be because the amount of meaningful information converges downward when the missing rate reaches a certain percentage in the data we used. Notwithstanding, our method imputes missing values more robustly than other methods when there is less information available (even at a 90% missing rate), which is consistent with the results in Chapter 3.

Addressing the class imbalance problem

Since the class imbalance problem not only leads to poor generalization [4], but is also related to poor accuracy as a result, we over-sampled data of minority class by the class-conditional generation that is considered as imputation of all features conditioned on the specific class. We verified that class conditional generation, adopted by our framework to deal with this issue, was performed successfully. We reduced the dimensionality of the hidden features ($\mathbf{H}$) and the input features ($\mathbf{s} = (\hat{\mathbf{t}}, \mathbf{h})$) of both the observed and synthesized data to two, using the t-SNE algorithm [102], and visualize their distributions. The distributions of the labeled hidden features $\mathbf{H}_l$ and the labeled input features $\mathbf{s}_l$ of observed data were changed as the model was trained because the encoders E_h and E_H and the generator G_{MI} were updated. We observed the distributions of the synthesized hidden features $\mathbf{H}_g$ and the synthesized input features $\mathbf{s}_g$ as the model was trained (Figure 5.5, columns 1-3). The distances between clusters of synthesized and observed data seemed to increase during some epochs, but the distributions of $\mathbf{H}_g$ and $\mathbf{s}_g$ eventually chased those of $\mathbf{H}_l$ and $\mathbf{s}_l$ (Figure 5.5, columns 4-5). The convergence of the distributions of the synthesized and observed values was seen to be accompanied by improved separation of the Aβ+ and A$\beta-$ groups. This suggests that our model accurately learned the data manifold occupied by the hidden features, and that it synthesized data that is realistic for each group. This solution to the class imbalance problem can be expected to improve the classification accuracy of the model.

Beyond missing values: imputing missing features

Figure 5.6(a) shows that the AUROC drops sharply, as expected, but AUROC remains above 0.74. Even when the fully trained model was tested with only MRI scans (Figure 5.6(a), green solid circle), it shows better performance than the model that was trained with only MRI scans (Figure 5.6(a), purple dashed line).

We also tested the ability of our model to make the same predictions when using different subsets of the features in inference. We used Rand indices to compare predictions of unlabeled data with all tabular features and with subsets of these features. We greedily removed the features which produced the largest drop in the Rand index. Predictions become less consistent as features are excluded, as we would expect. Nevertheless, only 12.3% of predictions changed when all the tabular features were excluded

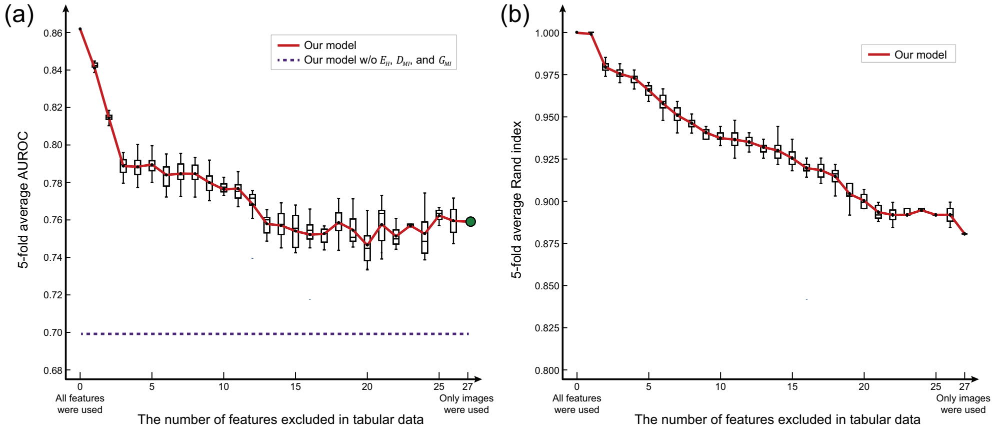


Figure 5.6: Inference using partial features. (a) The classification performance. This was measured using the average area under the receiver operating characteristic (AUROC) curves of trained models as features are removed one by one, measured by 5-fold cross validation. The average AUROCs over all possible combinations of features are shown as black dots, each within a box-and-whisker plot that represents the distributions of the AUROCs. The green solid circle shows the performance when a model trained on both tabular features and MRI scans was tested with only MRI scans. The purple dashed line shows the performance of a model trained on MRI scans alone. This model is similar to TripleGAN [95] but it performs oversampling in an embedded vector space as well as semi-supervised learning. (b) The ability of our model trained with both MRI scans and all tabular features to make the same prediction when using partial features in inference. This was evaluated using the Rand index [130] to 'reference' labels (predictions) of unlabeled data when all tabular features were used as features are removed one by one.

(Figure 5.6(b)).

Translation from discovery to practice datasets

It is desirable to use the rich information available in the discovery dataset from a large cohort study when we train the model with a practice dataset. However, both the feature sets and the diagnostic criteria found in the practice dataset from a specific clinical context can differ from those in a discovery dataset. The input features of the Aβ- groups overlap between these datasets to an extent; but the features of the Aβ+ groups in these datasets are significantly different (Figure 5.7(b), left plot). We would

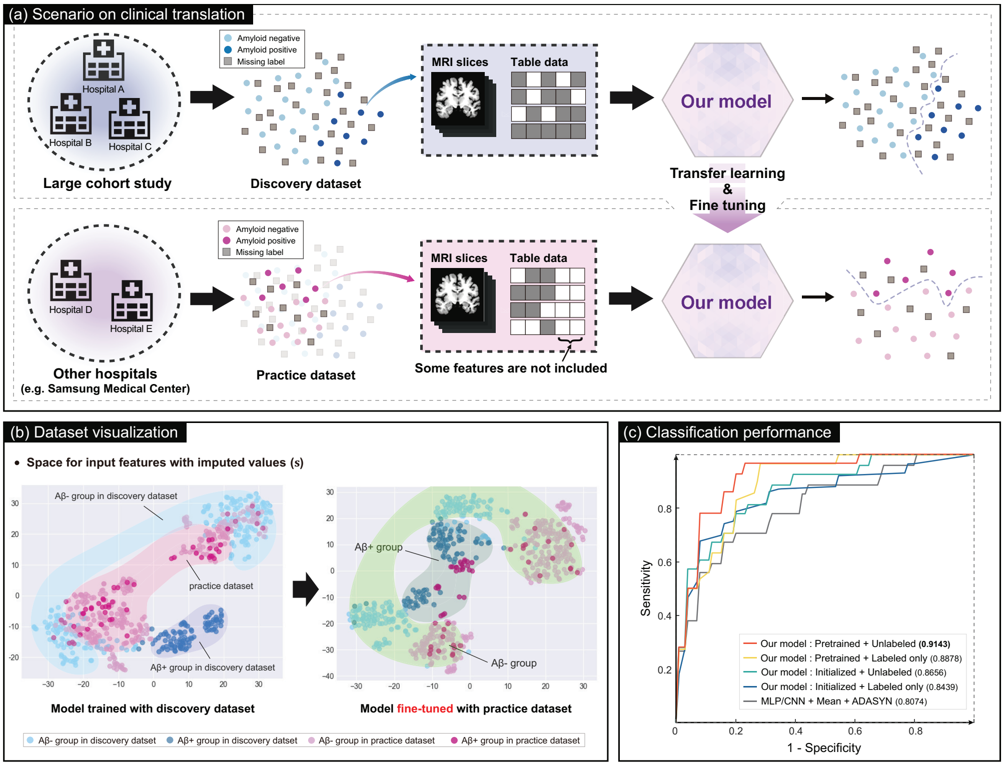

(a) Scenario on clinical translation
Hospital A
Hospital B
Hospital C
Large cohort study
Amyloid negative
Amyloid positive
Missing label
Discovery dataset
MRI slices
Table data
Our model
Transfer learning
&
Fine tuning
Hospital D
Hospital E
Other hospitals
(e.g. Samsung Medical Center)
Amyloid negative
Amyloid positive
Missing label
Practice dataset
MRI slices
Table data
Some features are not included
Our model
(b) Dataset visualization
• Space for input features with imputed values (s)
Aβ- group in discovery dataset
practice dataset
Aβ+ group in discovery dataset
Model trained with discovery dataset
Aβ+ group
Aβ- group
Model fine-tuned with practice dataset
Aβ- group in discovery dataset
Aβ+ group in discovery dataset
Aβ- group in practice dataset
Aβ+ group in practice dataset
(c) Classification performance
Sensitivity
1 - Specificity
0
0.2
0.4
0.6
0.8
Our model : Pretrained + Unlabeled (0.9143)
Our model : Pretrained + Labeled only (0.8878)
Our model : Initialized + Unlabeled (0.8656)
Our model : Initialized + Labeled only (0.8439)
MLP/CNN + Mean + ADASYN (0.8074)

Figure 5.7: Transferring information obtained from a large cohort study to specific clinical practice settings. (a) A model is trained from the discovery dataset obtained from the large cohort study. The practice dataset collected in another clinical setting may not include some tabular features present in the large cohort dataset. our model can synthesize values of those features that do not exist in a practice dataset, and can predict amyloid positivity using the model learned from the large cohort dataset. The discrepancy between the data distributions of the large cohort data and the practice dataset is addressed by transfer learning and fine-tuning. (b) When visualizing embedded vectors and imputed tabular data (**s**) for a model trained with the discovery dataset, the Aβ- and Aβ+ groups in the practice dataset are mixed with the Aβ- group from the discovery dataset. After the model has been fine-tuned with the practice dataset, the Aβ- and Aβ+ groups in the practice dataset are partially separated. (c) Average areas under the receiver operating characteristic (AUROC) curves for our model fine-tuned with the practice dataset and comparative models. Fine-tuning and semi-supervised learning improve performance.

not expect a model trained on the discovery dataset to make good predictions for the practice dataset. We therefore retrain our model by transfer learning and fine-tuning

(Figure 5.7(a)). This allows a model trained on the discovery dataset to synthesize features missing in the practice data due to not being collected by the relevant hospital, and then to use them to make predictions from the practice data.

We can visualize the labeled input features $\mathbf{s}_l$ by reducing their dimensionality using t-SNE. Looking at $\mathbf{s}_l$ obtained from the model trained with the discovery dataset, the Aβ- and Aβ+ groups in the practice datasets are not clearly differentiated. After fine-tuning the model with the practice dataset, the separation of the Aβ- and Aβ+ groups in the practice dataset is much clearer (Figure 5.7(b), right plot), indicating that the model has been adjusted to the practice dataset.

After fine-tuning, our model achieved an average AUROC of 0.9143, an average accuracy of 0.8528, an average sensitivity of 0.9667, and an average specificity of 0.8286 at optimal thresholds determined by the Youden index (Figure 5.7(c), red line). The effectiveness of fine-tuning is demonstrated by the significantly worse performance of a model trained from randomly initialized weights (Figure 5.7(c), cyan line). Our method also outperforms other deep learning models with machine learning techniques designed for imperfect datasets (Figure 5.7(c), gray line). We can also see that semi-supervised learning improves the prediction performance (Figure 5.7(c), red versus yellow and cyan versus blue).

Discriminative regions and variables related to amyloid positivity

The three most discriminative regions of amyloid positivity exist in the right posterior temporal lobe, and in the right and left lateral remainders of the occipital lobes (Figure 5.8(a) and (b)). The most discriminative tabular features are sex and the number of APOE ϵ4 alleles (Figure 5.8(c)).

When our model is trained using fine-tuning with the practice dataset, the most discriminative regions or variables were consistent with those with the discovery dataset. However, the significance patterns were somewhat different from each other. The top three discriminative regions are the left and right lateral remainders of the occipital lobes, and the brainstem (Figure 5.8(d) and (e). Again, the most discriminative tabular features are sex, and the number of APOE $\epsilon4$ alleles are highly discriminative of amyloid positivity. The most discriminative tabular feature is the number of APOE $\epsilon4$ alleles (Figure 5.8(f)).

We also compare various methods to calculate the importance of brain regions and tabular features. The methods we compare include the mean-square importance that we used, the mean importance, the square-mean importance, and the absolute-mean importance as follows:

$$\text{The mean-square importance: } \mathcal{I}_{r_i}^n = \left(\frac{1}{n_{r_i}}\sum_{v \in r_i} A_{I_v}\right)^2, \ \mathcal{I}_{t_i}^n = (A_{t_i}^n)^2 \tag{5.35}$$

$$\text{The mean importance: } \mathcal{I}_{r_i}^n = \frac{1}{n_{r_i}}\sum_{v \in r_i} A_{I_v}, \ \mathcal{I}_{t_i}^n = A_{t_i}^n \tag{5.36}$$

$$\text{The square-mean importance: } \mathcal{I}_{r_i}^n = \frac{1}{n_{r_i}}\sum_{v \in r_i} (A_{I_v})^2, \ \mathcal{I}_{t_i}^n = (A_{t_i}^n)^2 \tag{5.37}$$

$$\text{The absolute-mean importance: } \mathcal{I}_{r_i}^n = \frac{1}{n_{r_i}}\sum_{v \in r_i} |A_{I_v}|, \ \mathcal{I}_{t_i}^n = |A_{t_i}^n| \tag{5.38}$$

Figures 5.9, 5.10, 5.11, and 5.12 show the average importance values over Aβ+ or Aβ- participants on the discovery dataset.

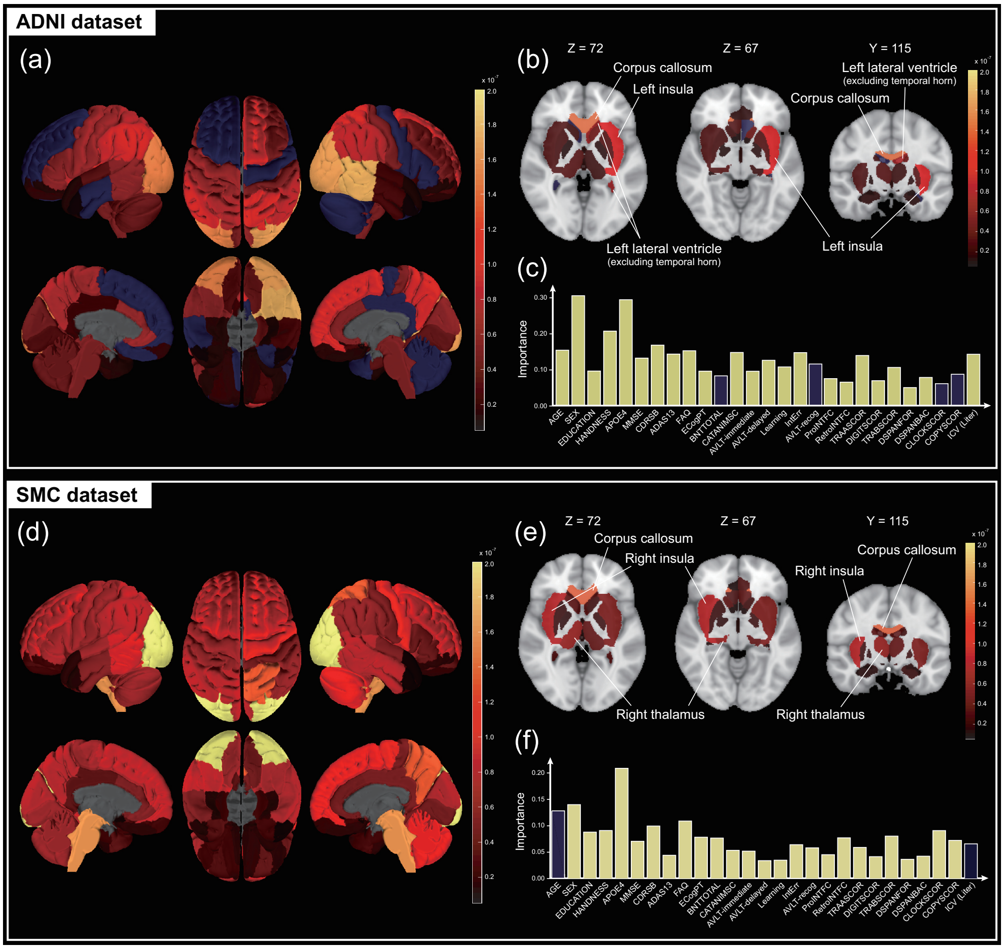
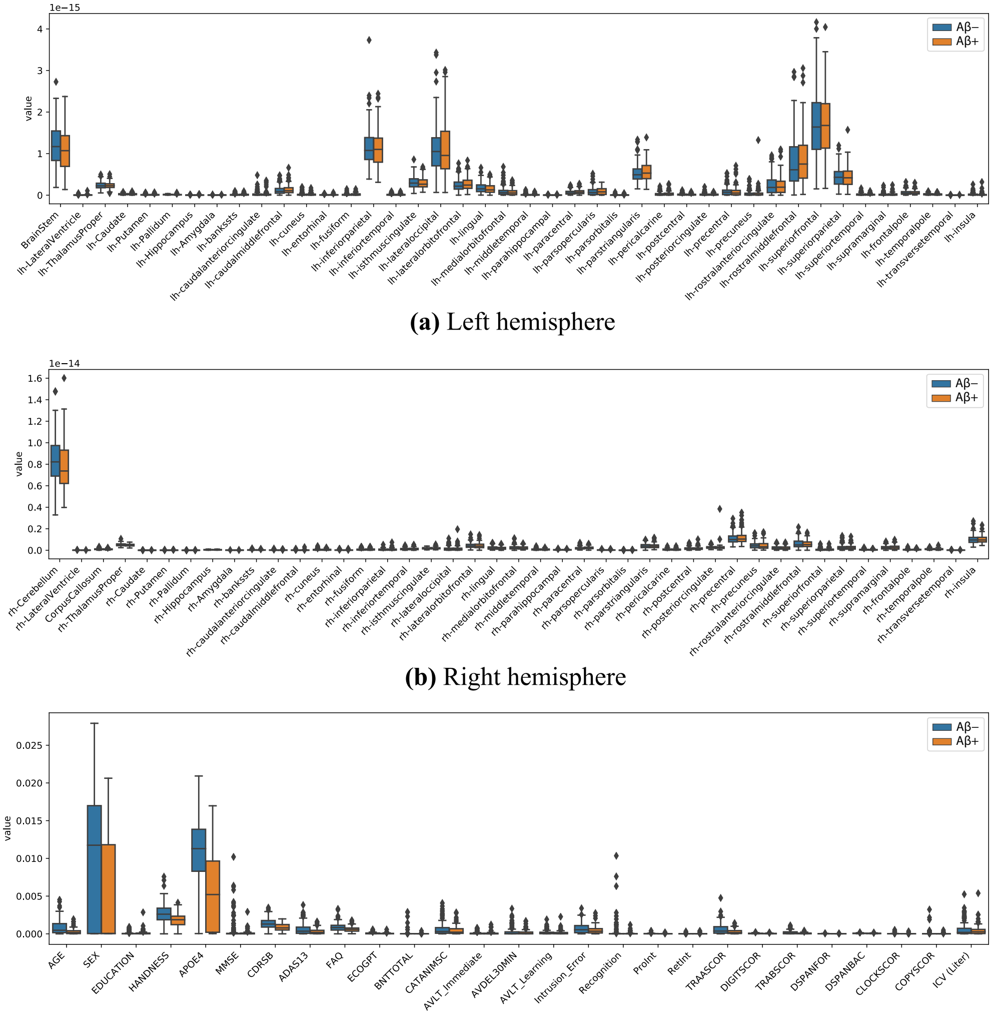
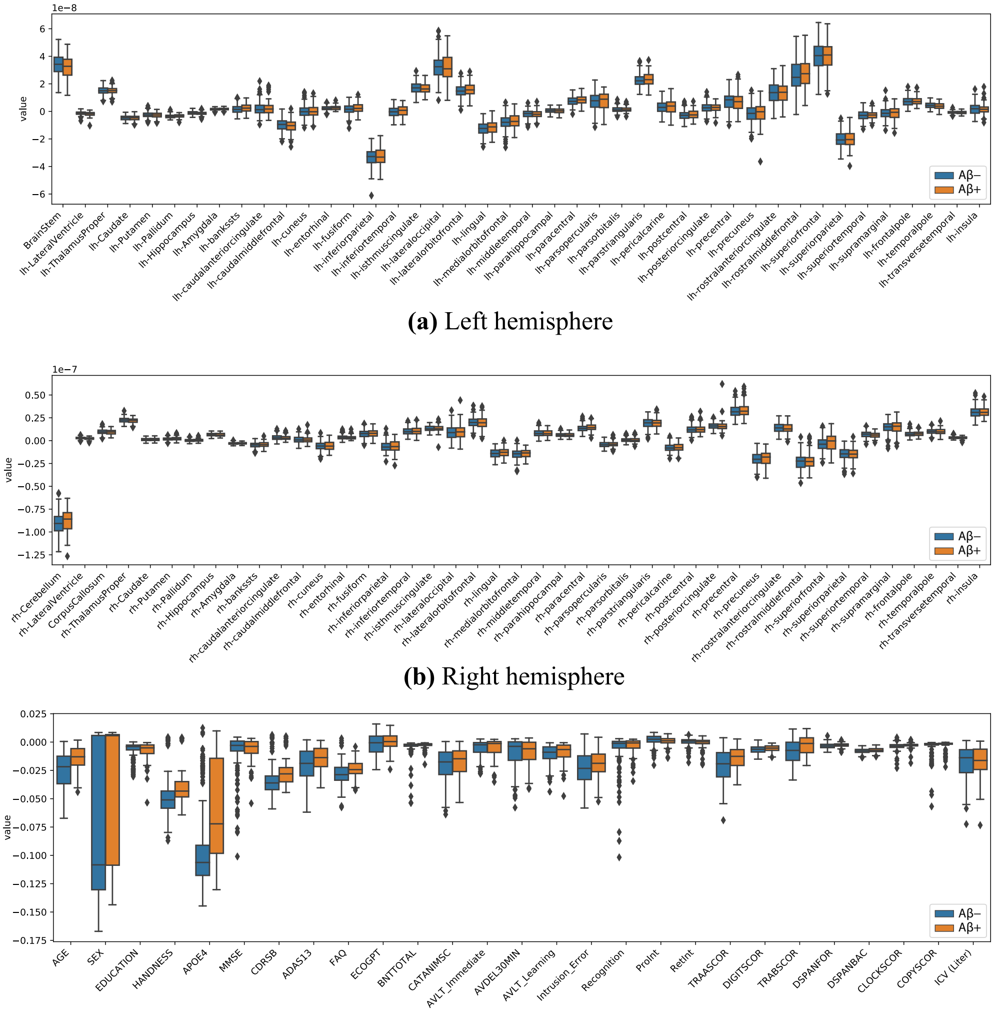


Figure 5.8: Discriminative regions and variables of amyloid positivity determined by our model trained with the discovery and practice datasets. (a) Brain maps for interpreting how our model trained with the discovery dataset predicted amyloid positivity. Importance values are expressed according to the Hammersmith (HM) [62] atlas. Color scale represents importance values computed using the integrated gradient (IG) [145] method. Dark blue regions make no significant contribution to classifying amyloid positivity. (b) Discriminative regions that cannot be shown on the cortical surface. (c) Discriminative features: those colored dark blue make no significant contribution to classifying amyloid positivity. (d)-(f) Discriminative regions and variables determined by our model, fine-tuned with the practice dataset. Figure courtesy of Sung-Woo Kim.



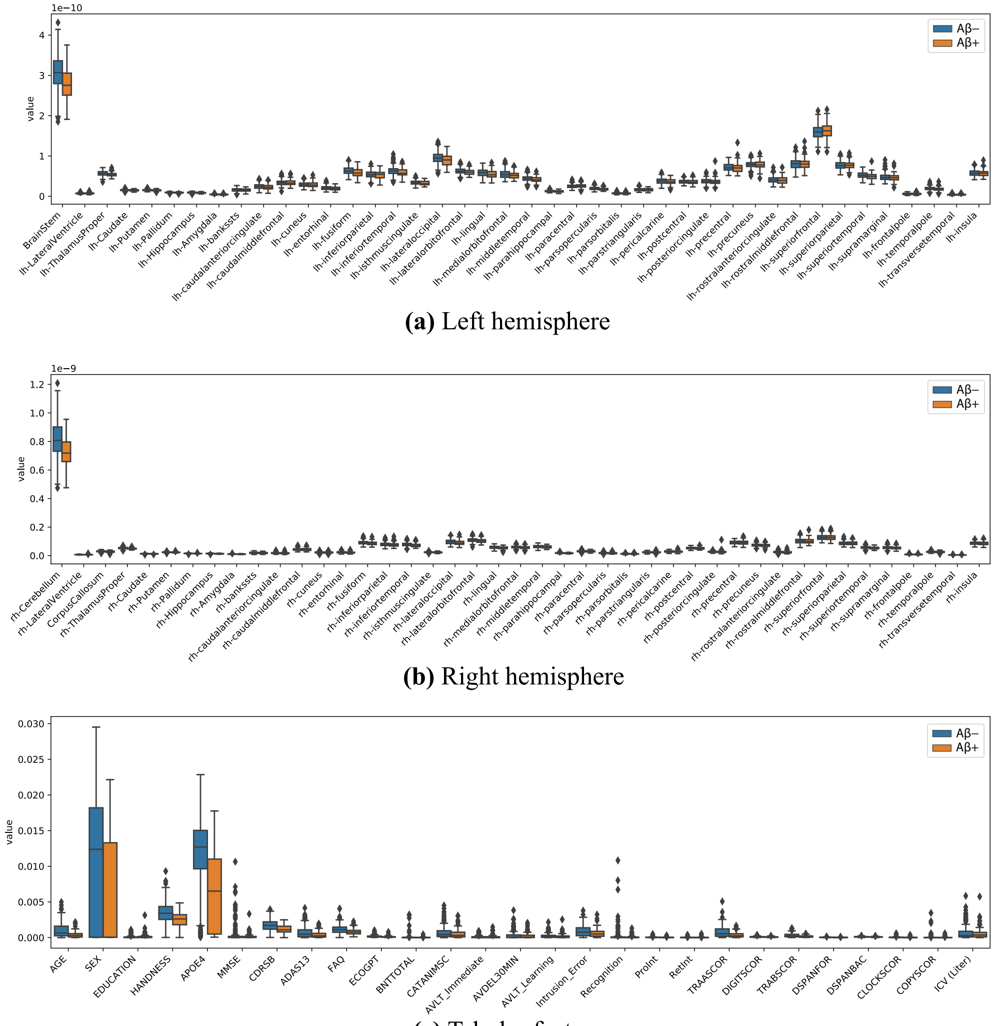


(a) Left hemisphere

(b) Right hemisphere

(c) Tabular features

Figure 5.10: The average mean importance

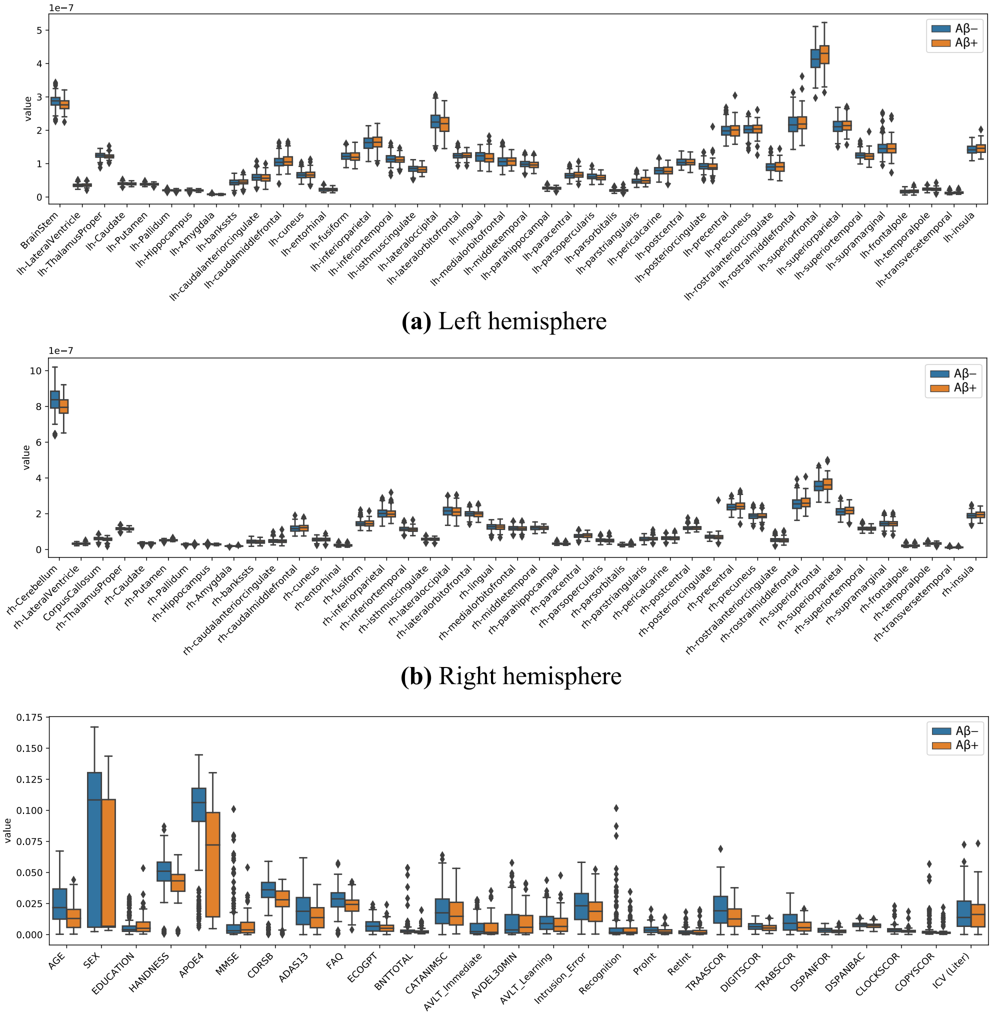


(a) Left hemisphere

(b) Right hemisphere

(c) Tabular features

Figure 5.11: The average square-mean importance



(a) Left hemisphere

(b) Right hemisphere

(c) Tabular features

Figure 5.12: The average absolute-mean importance

5.1.5 Discussion

We have proposed a deep learning technique for predicting the preclinical stage of Alzheimer's disease in cognitively normal individuals from structural magnetic resonance imaging scans, demographic variables and various cognitive scores. Our model can cope with real-world situations in which a) the training dataset is imperfect, b) the feature set of the test data is a subset of the features used in training, and c) different labels are used in the test and training data. Our deep generative model overcomes these real-world problems in the clinical implementation of deep learning for the determination of early Aβ pathology by implicitly estimating a joint distribution of features and outcomes [70]. We also show that non-linear discriminative regions can be used to explain how our model allocates data to Aβ+ or Aβ-.

The HexaGAN framework successfully addressed the imperfect dataset problem in terms of 'imputation' by creating additional realistic data: substitutes for missing values were created by optimizing element-wise adversarial loss between a generator G_{MI} and a discriminator D_{MI}. This is shown to be more accurate than previous competitive techniques; pseudo-labels to replace missing labels were generated using a classifier C, which is trained with the discriminator D_{MI} in adversarial fashion; and instances of the minority class were oversampled by the class-conditional generation. Our model based on the HexaGAN framework efficiently utilizes imperfect data and, therefore, effectively predicts diagnoses in a real-world situation while interplaying between components to solve these sub-problems simultaneously, not separately in a different order.

The imputation of missing data is based on the data distribution. However, it is difficult to estimate the distribution of high-dimensional features of MRI scans together with tabular data. Thus, we reduce the dimensionality of this distribution. Firstly, we selected 18 slices from the total of 210 coronal slices of MRI scans. The prediction performance of our model with only a small number of slices rather than the whole image indicates that deep learning can be applied to predict the preclinical stage of AD using MRI scans acquired in clinical practice. Secondly, we introduced the encoder E_h into HexaGAN to extract low-dimensional embedded vectors $\mathbf{h}$ from the MRI scans. Our model can oversample embedded vectors $\mathbf{h}$ with hidden features from the image data, which makes imputation much more straightforward, rather than a high-dimensional MRI scan per se which could depress the classification performance by the curse of dimensionality [47].

The prediction performance of our model trained with the discovery dataset was the AUROC of 0.8609 ± 0.0266 (Figure 5.2), which was higher than previous models trained with cross-sectional structural MRI scans [92, 124, 151, 156], and comparable with a model trained on longitudinal structural MRI scans [125]. We attribute this performance improvement to the capability of DL models to find non-linear relationships between features and predictions. In our model, this capability is increased by using all observed data efficiently via imputation, which increases the number of data available for training. This theoretically brings down the upper bound on the difference between the generalization error and the empirical error [170]. In addition, reducing the dimensionality of the MRI scan is a pragmatic way to reduce the size of the problem and to make the predictions of our model more comprehensible. These features of our model are combined with deep generative models and appear to give better results than previous studies [3] with the currently feasible capability of DL models.

Figure 5.2(a) suggests that the predictions made by our model were most accurate when both image and tabular data were used as features. The accuracy dropped when only tabular data was used and dropped further when only image data was used. This suggests that the encoder E_h is effective in extracting features from reduced-dimensionality images obtained from MRI scans, although the tabular features are more important. In addition, E_H of our model combined the essential information of image data with tabular data and exerted a synergistic effect on the performance of our model. On the other hand, the performance of discriminative DL models such as multilayer perceptrons (MLP) or convolutional neural networks (CNN) was not much affected by the omission of the image data. This is because CNNs are poor at extracting information from image data [47].

We would expect our network to be applicable to many clinical situations, in which both the available features and their labeling differ from those in the dataset used to train the model (Figure 5.7(a)). Our model can impute values for features absent from the practice dataset by estimating the class-conditional density $p(\mathbf{H}|y)$. The effectiveness of this is shown in Figure 5.6(a). The problem of inconsistent labeling can be addressed by fine-tuning. Although fine-tuning confines the search area to learn the model for new data, models can employ information from the large data inherent in pretrained models [121]. Several studies corroborated that fine-tuning is also effective in constructing prediction models for biomedical data [131, 168]. In addition, fine-tuning can also be beneficial for generative models when the size of a training dataset is limited to generate synthetic data [164]. Figure 5.7(b) shows that fine-tuning is successful in adjusting our model to the practice dataset: the difference in the class-conditional density $p(\mathbf{s}|y)$ between the discovery dataset and the practice dataset is reduced, and the model's predictions are more accurate (Figure 5.7(c)). Fine-tuning with the practice dataset had little effect on the discriminative profile of either the image or the tabular data (Figure 5.8).

Explanable AI is a method that allows humans to understand the outputs generated by machine learning models. It helps to characterize model accuracy and fairness and describes the expected effects and potential biases of the model. Individual-level explainability describes the prediction of an instance created by the model. This may provide an understanding of each individual (e.g., false positives or false negatives). Population-level explainability is measured by pooling information on individual-level explainability. This provides abstraction and summary of the model's decision boundary through discriminative variables and provides the model's reliability within the comprehension capability of humans (e.g., physicians).

DL models are known to be difficult to make or verify hypotheses. Nevertheless, it is often valuable to know how the features influenced a prediction. Discriminative regions and variables can help physicians understand model predictions and diagnose diseases, while also providing reliability to patients. We can determine discriminative regions and variables for the model trained with the discovery dataset, with or without fine-tuning on the practice dataset, using importance values obtained from the integrated gradient (IG) algorithm [145]. We believe that this is the first study in which XAI techniques have been applied to multi-modal (image and table) data and provided discriminative regions and features for the entire dataset as well as for each participant. The most discriminative regions including posterior regions of the brain and the brainstem are among those associated with late stages of Aβ pathology [19, 55, 119], and overlap with previous classifications [22, 34]. One explanation of the high discriminative power of these regions is that the criteria for amyloid positivity employed in positron emission tomography (PET) tend to favor the later stages of amyloid deposition, more than cerebrospinal fluid (CSF) measures [119]. We note that this discriminative power may not be a direct result of amyloid deposition, but a concomitant alteration of texture or shape [34]. For the tabular features, the association between the presence of APOE $\varepsilon 4$, the most discriminative variable in our model, and Aβ deposition has been previously reported [91, 134, 151]. Sex, an important discriminator in our network trained on the discovery dataset alone, is known to be a factor in AD [10, 96, 105], but the discriminative power of this feature may be, in part, influenced by a characteristic of the dataset, which is skewed towards female participants (Table 5.1). Our ablation studies (Figure 5.6(a)) also suggested that these two features were the most discriminative.

There are some limitations on our model. At present, an MRI scan is always required; but we plan to extend our method to allow inference from tabular data alone. Secondly, it is not clear whether the importance values that are used to interpret model predictions arise from the morphological characteristics of the brain, such as cortical thickness or cortical widening, or from the overall intensity of the MR signal. We leave this avenue of research for future work.

The content contained within this chapter, including passages, figures, and tables, may appear in future published papers.

5.1.6 Appendix

Implementation details

Table 5.7: Network architectures.

E_h	E_H	G_{MI}	D_{MI}
$I \in \mathbb{R}^{168\times190\times18}$	$(\tilde{\mathbf{t}}, \mathbf{m}) \in \mathbb{R}^{27+27}$	$\mathbf{H} \in \mathbb{R}^{256}$	$\mathbf{t} \in \mathbb{R}^{27}$
DenseNet121	FC(512) + SN + ReLU	FC(512) + SN + ReLU	FC(2048) + SN + LReLU
FC(2048) + Sigmoid	FC(2048) + SN + ReLU	FC(512) + SN + ReLU	Concat($\mathbf{h}$) $\in \mathbb{R}^{2048+2048}$
	Concat($\mathbf{h}$) $\in \mathbb{R}^{2048+2048}$	FC(512) + SN + ReLU	FC(512) + SN + LReLU
	FC(512) + SN + ReLU	FC(512) + SN + ReLU	FC(512) + SN + LReLU
	FC(512) + SN + ReLU	FC(512) + SN + ReLU	FC(512) + SN + LReLU
	FC(256) + Tanh	FC(512) + SN + ReLU	FC(512) + SN + LReLU
		FC(27+2048) + Sigmoid	Projection(y, 27+2048+2) + Sigmoid
G_{CG}	D_{CG}	C	
$(\mathbf{z}, y) \in \mathbb{R}^{27+2}$	$(\mathbf{H}, y) \in \mathbb{R}^{256+2}$	$\mathbf{t} \in \mathbb{R}^{27}$	
FC(512) + SN	FC(512) + SN + LReLU	FC(2048) + ReLU + Dropout(0.5)	
FC(512) + SN	FC(512) + SN + LReLU	Concat($\mathbf{h}$) $\in \mathbb{R}^{2048+2048}$	
FC(512) + SN	FC(512) + SN + LReLU	FC(512) + ReLU + Dropout(0.5)	
FC(256) + Tanh	FC(512) + SN + LReLU	FC(512) + ReLU + Dropout(0.5)	
	Projection(y, 1) + Sigmoid	FC(2) + Softmax	

FC(m): Fully-connected layer with hidden dimension m.
SN: Spectral normalization [113].
DenseNet121: DenseNet model [67] pretrained on the ImageNet dataset.
ReLU, LReLU, Tanh, Sigmoid, Softmax: Activation functions
Concat(a): The output of the prvious layer is concatenated with a
Dropout(p): Output neurons are randomly turned off with probability p [48]
Projection(y, d): Projection layer [111] to inject the conditional information of y with output dimension d

Table 5.7 shows network architectures used in our experiments. Each component of whole system is updated in order. Since the distribution of H is altered by updating E_H and E_h, we update G_{CG} and D_{CG} 30 times for each update of the other components, in order to estimate the distribution of H accurately. That is, 30 × $(G_{\text{CG}} \to D_{\text{CG}}) \to E_H \to G_{\text{MI}} \to D_{\text{MI}} \to E_h \to C$.

We adapted HexaGAN to deal with problems derived from the size of MRI data in two ways. Firstly, we added an additional encoder network E_h in front of HexaGAN to reduce the dimensionality of the data. To train the encoder E_h more efficiently, its training is augmented by transfer learning using DenseNet with parameters pre-trained on ImageNet [67, 121] except the last two layers. We applied spectral normalization to the weights of every layer of E_H, D_{CG}, G_{CG}, D_{MI}, and G_{MI} except output layers of E_H, G_{CG}, and G_{MI}, to stabilize the learning process [113]. Further, we added projection discriminators to D_{CG} and D_{MI}, in order to reinforce class conditioning of the GAN [111].

We implemented deep learning networks using the Tensorflow library [1]. We used the Adam optimizer to update the networks with the aim of minimizing loss functions through back-propagation. We used the Adam optimizer with $\beta_1 = 0$ and $\beta_2 = 0.9$. The learning rate of C was set to 0.0001, and those of D_{CG} and D_{MI} were set to 0.00016 by applying the two time-scale update rule (TTUR) [66], and that of the other components were set to 0.00004. We used one Nvidia TITAN V for training the model.

We calculated test AUROCs for all epochs during model training and stored the model parameters at the epochs that show the highest test AUROCs at each fold. We averaged these five test AUROCs to produce a final classification performance. The CNNs in comparison models are trained in a similar way to ours, by transfer learning from DenseNet parameters pre-trained on ImageNet. We also set their hyperparameters, such as the learning rate, to the same values that we used in our model.

5.2 Prediction of Mortality and Intervention in COVID-19 Patients

5.2.1 Introduction

The coronavirus disease 2019 (COVID-19) has spread around the world since December 2019. The explosion in the number of patients causes a shortage of medical resources including medical staffs and hospital beds [6], so accurate screening of patients who are at high risk of death or who require medical interventions helps efficient allocation of medical resources. In addition, timely clinical interventions, such as intubation and supplemental oxygen, are important to reduce inpatient mortality.

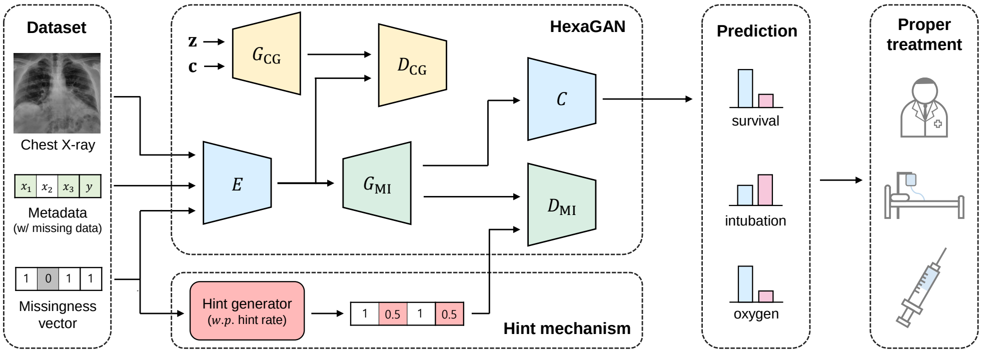


Figure 5.13: Overview of prediction of mortality and intervention in COVID-19 patients using GANs.

However, it is difficult for medical staffs to make decisions about treatment in real time and to allocate medical resources efficiently due to the large number of COVID-19 patients. This has increased the demand for automation of the decision-making process using patients' electronic health records (EHR). To meet the demand, automated tools, especially deep learning (DL) methods, to predict the risk of death and interventions have been actively proposed [137, 11, 161].

Although DL models for classification and prediction have remarkably advanced [67, 149], limited data problems in EHRs have stunted the application of the DL-based methods. There are three typical limited data problems in EHRs. First, some attributes can be missing (missing data). Second, outcomes can be missing due to the labeling cost (missing label). Third, outcomes can be imbalanced (class imbalance). For these reasons, DL models for predicting various outcomes related to COVID-19 may not perform best.

In order to address these problems, machine learning techniques for imputation, oversampling, and semi-supervised learning can be applied. Recently, deep generative models for limited data problems have been proposed [136, 95, 171]. Hwang et al. [70] suggested a unified view of limited data problems via imputation and proposed generative adversarial networks (GANs) that can address the three problems simultaneously.

In this chapter, we used a publicly available dataset [30] of chest X-ray images and metadata to predict mortality and interventions of COVID-19 patients. We adopted HexaGAN [70] and additionally applied a hint mechanism [171] to accurately predict the mortality, and whether the patients need intubation or supplemental oxygen. We verified that our method outperforms combinations of existing techniques for limited data problems.

The contributions are summarized as follows:

- We construct deep generative models to provide accurate prediction of mortality and interventions for COVID-19 patients from the dataset with limited data problems.

- To enable this, we use generative adversarial networks named HexaGAN which addresses limited data problems simultaneously and additionally apply a hint mechanism to enhance the prediction performance.

- Our method significantly outperforms combinations of existing methods. Especially, our method achieves about twice higher performance (specificity) than benchmark combinations in predicting the risk of death.

5.2.2 Proposed method

We used the HexaGAN framework to address limited data problems and applied a hint mechanism to enhance the prediction performance. The overview of our method is illustrated in Figure 5.13.

HexaGAN

For our task, the dataset contains chest X-ray image data $\mathbf{I}$, tabular metadata $\mathbf{t}$, and outcomes $\mathbf{y}$. We construct a boolean vector $\mathbf{m}$ named a missingness vector to indicate whether an element of metadata is missing. If i-th element of metadata is missing, m_i is 0, and vice versa. Since the dataset has the missing data, class imbalance, and missing label problems simultaneously, we adopted the HexaGAN framework [70] which is the state of the art method for this scenario.

HexaGAN consists of six components including an encoder, generators, discriminators, and a classifier. These components interact with each other via adversarial learning and low-dimensional vectors $\mathbf{h}$ in the hidden space. Role of each component is summarized as follows:

- The encoder E receives $\mathbf{I}$, $\mathbf{t}$, and $\mathbf{m}$ and synthesizes a low-dimensional hidden vector $\mathbf{h}$ in the hidden space.
- A generator for missing imputation G_{MI} receives $\mathbf{h}$ and performs missing data imputation and oversampling.
- A discriminator for missing imputation D_{MI} receives imputed data and predicts $\mathbf{m}$.
- A generator for conditional generation G_{CG} receives a minority class label $\mathbf{c}$ and a noise vector $\mathbf{z}$ and generates $\mathbf{h}$ to oversample data in the minority class.
- A discriminator for conditional generation D_{CG} distinguishes between $\mathbf{h}$ from real data and $\mathbf{h}$ synthesized by G_{CG}.
- The classifier C provides predictions given imputed data and generates pseudo-label of unlabeled data to perform semi-supervised learning.

As described in Chapter 3, these components interact with each other to address the limited data problems. Each component of the whole framework is updated in rotation (See Algorithm 3.2).

Hint mechanism

A hint mechanism, which is proposed by Yoon et al. [171], is a random variable containing partial information about missingness. We used the hint mechanism to improve

Table 5.8: Description of metadata in the dataset

Attribute	Description	Data type	Missing rate
age	Age	integer	27.2%
offset	Days elapsed since the onset of symptoms or hospitalization	integer	23.5%
sex	Male or female	binary	7.5%
RT-PCR positive	Yes or no	binary	41.1%
went icu	Whether the patient was in the intensive care unit or critical care unit	binary	54.7%
extubated	Whether the patient was successfully extubated	binary	95.6%
temperature	Temperature of the patient (°C)	continuous	91.7%
pO2 saturation	Partial pressure of oxygen saturation (%)	continuous	86.3%
wbc count	White blood cell count (10^3/uL)	continuous	98.2%
neutrophil count	Neutrophil cell count (10^3/uL)	continuous	96.7%
lymphocyte count	Lymphocyte cell count (10^3/uL)	continuous	95.6%
survival	Yes or no	binary	65.8%
intubated	Whether the patient was intubated (or ventilated)	binary	67.5%
supplemental O2	Whether the patient required supplemental oxygen	binary	87.5%

Table 5.9: The number of outcomes with imbalance ratios and missing rates

Outcome	Y	N	Missing	Imb. ratio (1:x)	Missing rate
survival	162	38	384	4.3	65.8%
intubated	114	76	394	1.5	67.5%
supplemental O2	53	20	511	2.7	87.5%

the imputation performance of our method, which eventually enhanced the prediction performance of the classifier. When the hint rate is p which is the hyperparameter, each element of a hint mechanism $\mathbf{H}$ is sampled as follows:

$$H_i = \begin{cases} m_i \quad w.p. \;\; p \\ 0.5 \quad w.p. \;\; 1 - p \end{cases} \tag{5.39}$$

A sampled $\mathbf{H}$ is fed into D_{MI}. With little computational overhead, it helps D_{MI} predict $\mathbf{m}$, and improve the imputation performance of our method.

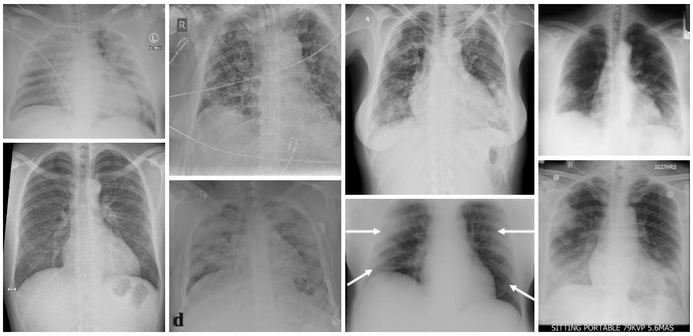


Figure 5.14: Samples of X-ray image data of COVID-19 patients.

5.2.3 Experimental setup

Datasets

We used the covid-chestxray-dataset [30] for the prediction of mortality and medical interventions. The dataset includes 846 records who have been confirmed or suspected of COVID-19 or other viral and bacterial pneumonias. Among them, 468 records were obtained from COVID-19 patients, and those with outcome attributes were used as labeled data. Records obtained from other pneumonias patients were treated as unlabeled data regardless of the missingness of outcome attributes. We used X-ray image data and metadata which contain the patients' information. The descriptions of metadata attributes are shown in Table 5.8. We used the bottom three attributes as outcomes. For example, we used *survival* to predict mortality, *intubated* to predict intubation, and *supplemental O2* to predict supplemental oxygen. The samples of X-ray image data are shown in Figure 5.14, and the number of outcomes and missing rates are presented in Table 5.9.

Evaluation metrics

We computed the F1-score, the area under the receiver operating characteristic curve (AUROC), the sensitivity, and the specificity as the evaluation metrics. We used 5-fold cross validation and reported the average of the test performance for each fold.

5.2.4 Experimental results

Mortality prediction

Benchmark combinations To find the most competent combinations of existing techniques for limited data problems, we conducted extensive experiments on mortality prediction as shown in Table 5.10. We used mean imputation (Mean), k-nearest neighbors (kNN) [157], and multivariate imputation by chained equations (MICE) [21] for the missing data problem; synthetic minority oversampling technique (SMOTE) [24],

Table 5.10: Mortality prediction performance of combinations of existing techniques

Missing data	Class imbalance	Missing label	F1-score	Accuracy	AUROC	Sensitivity	Specificity
Mean	SMOTE	kNN	0.9162	0.8570	0.8490	0.9748	0.3909
kNN	SMOTE	kNN	**0.9411**	**0.8948**	**0.8893**	0.9861	0.3748
MICE	SMOTE	kNN	0.9198	0.8535	0.8093	0.9972	0.0826
Mean	ADASYN	kNN	0.8992	0.8251	0.8266	0.9778	0.2232
kNN	ADASYN	kNN	0.9367	0.8854	0.8888	0.9945	0.2631
MICE	ADASYN	kNN	0.9159	0.8464	0.7736	0.9930	0.0595
Mean	CRL	kNN	0.8961	0.8180	0.7888	0.9837	0.1642
kNN	CRL	kNN	0.9333	0.8794	0.8067	0.9847	0.2729
MICE	CRL	kNN	0.9181	0.8522	0.7612	0.9818	0.1567
Mean	CS	kNN	0.9029	0.8321	0.7860	0.9763	0.2634
kNN	CS	kNN	0.9213	0.8546	0.7193	**1.0000**	0.0240
MICE	CS	kNN	0.9162	0.8475	0.7100	0.9874	0.0963
Mean	SMOTE	LP	0.8713	0.7778	0.7781	0.9815	0.1119
kNN	SMOTE	LP	0.8762	0.7920	0.8163	0.9905	0.2201
MICE	SMOTE	LP	0.8839	0.8085	0.8307	0.9610	0.3317
Mean	ADASYN	LP	0.9015	0.8381	0.8036	0.9676	**0.4141**
kNN	ADASYN	LP	0.8739	0.7932	0.8132	0.9666	0.2926
MICE	ADASYN	LP	0.8746	0.7836	0.7914	0.9953	0.1220
Mean	CRL	LP	0.8740	0.7790	0.7525	**1.0000**	0.0558
kNN	CRL	LP	0.8744	0.7943	0.7886	0.9569	0.3231
MICE	CRL	LP	0.8739	0.7943	0.8164	0.9392	0.3415
Mean	CS	LP	0.8906	0.8144	0.7579	0.9815	0.2683
kNN	CS	LP	0.8718	0.7825	0.7625	0.9921	0.1793
MICE	CS	LP	0.8751	0.7896	0.8287	0.9719	0.2195

adaptive synthetic (ADASYN) [64], class rectification loss (CRL) [38], and cost sensitive loss (CS) [144] for the class imbalance problem; kNN and label propagation (LP) [180] for the missing label problem. As a result, a combination of kNN for the missing data problem, SMOTE for the class imbalance problem, and kNN for the missing label problem shows the highest performance in terms of the F1-score, accuracy, and AUROC. A combination of Mean for the missing data problem, ADASYN for the class imbalance problem, and LP for the missing label problem shows the highest performance in terms of specificity. Specificity is an important performance metric for mortality prediction. Since the outcome is 1 when the patient is alive and the outcome is 0 when the patient is dead, the specificity becomes the probability of correctly identifying patients who have died, which is an important indicator for accurately predicting patients who need urgent treatment. Therefore, we chose these two combinations as competent benchmark combinations to compare with our method.

Table 5.11: Performance comparison on mortality prediction

Method	Hint rate	F1-score	AUROC	Sensitivity	Specificity
kNN + SMOTE + kNN	N/A	0.9411	0.8893	0.9861	0.3748
Mean + ADASYN + LP	N/A	0.9015	0.8036	0.9676	0.4141
HexaGAN	0	0.9566	0.9395	**1.0000**	0.6571
HexaGAN w/ hint	0.7	**0.9659**	**0.9508**	**1.0000**	0.7393
HexaGAN w/ hint	0.8	0.9610	0.9464	0.9714	**0.8071**

Effectiveness of hint Figure 5.15 shows the benefit of a hint mechanism on mortality prediction. We evaluated the performances of our method by changing hint rate from 0 to 0.9. If the hint rate is 0, it is the same as HexaGAN without a hint mechanism. The hint mechanism improved the prediction performance of HexaGAN. If the hint rate is too high, the number of elements available for training becomes insufficient, impairing the learning process. F1-score is best at hint rate 0.7, and specificity is best at hint rate 0.8. Therefore, we chose these two hint rates to conduct comparative studies.

Comparison on mortality prediction Table 5.11 compares the performance of our method with benchmarks on mortality prediction. Our method (HexaGAN with a hint mechanism) outperformed benchmark combinations. Especially, our method with a hint rate of 0.8 performed approximately twice better than benchmark combinations in terms of specificity, indicating that it predicted critically ill patients approximately two times better.

Intervention prediction

We also predicted medical interventions including intubation and supplemental oxygen using our method, and conducted comparative studies between our method and benchmark combinations.

Comparison on intubation prediction Table 5.12 compares the performance of our method with benchmark combinations on intubation prediction. We did not use

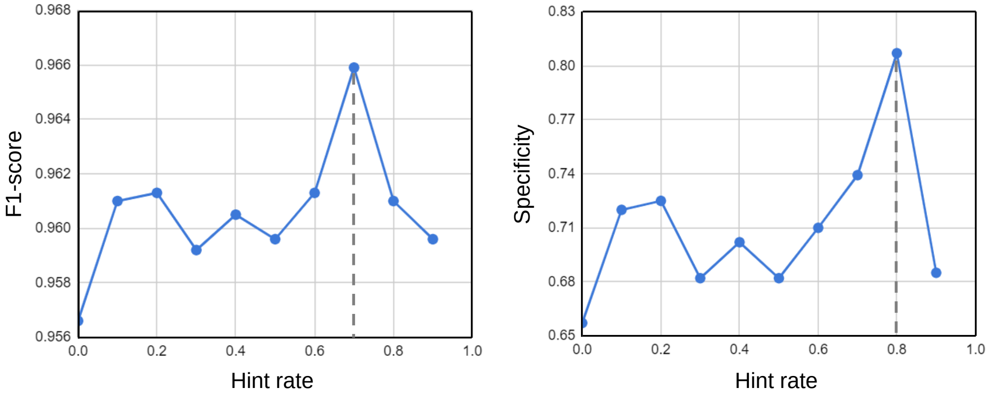


Figure 5.15: Performance comparison with respect to the hint rate.

Table 5.12: Performance comparison on intubation prediction

Method	Hint rate	F1-score	AUROC	Sensitivity	Specificity
kNN + SMOTE + kNN	N/A	0.9513	0.9490	0.9805	0.7826
Mean + ADASYN + LP	N/A	0.7133	0.6115	0.9742	0.0841
HexaGAN	0	0.9563	0.9729	**1.0000**	0.8227
HexaGAN w/ hint	0.7	0.9604	**0.9771**	**1.0000**	0.8409
HexaGAN w/ hint	0.8	**0.9648**	0.9764	**1.0000**	**0.8591**

Table 5.13: Performance comparison on supplemental O2 prediction

Method	Hint rate	F1-score	AUROC	Sensitivity	Specificity
kNN + SMOTE + kNN	N/A	0.9537	0.9771	0.9459	0.7493
Mean + ADASYN + LP	N/A	**0.9730**	0.9713	0.9744	0.8000
HexaGAN	0	0.9513	**1.0000**	**1.0000**	0.7500
HexaGAN w/ hint	0.7	0.9609	**1.0000**	**1.0000**	0.8000
HexaGAN w/ hint	0.8	0.9568	**1.0000**	0.9556	**0.9000**

the *extubated* attribute as a predictor for intubation prediction, because a patient can only be extubated if they were intubated at some point. The combination of Mean + ADASYN + LP failed to predict the intubation of patients. Our method showed the best performance across all evaluation metrics. HexaGAN with hint rates 0, 0.7, 0.8 showed a sensitivity of 1.0000, which indicates that our method predicts all patients who are actually intubated as patients in need of intubation.

Table 5.14: Effect of semi-supervised learning. SSL is whether unlabeled data are used or not.

Outcome	SSL	Hint rate	F1-score	AUROC	Sensitivity	Specificity
survival	✗	0.7	0.9590	0.9428	0.9928	0.7035
	✓	0.7	**0.9659**	**0.9508**	**1.0000**	**0.7393**
intubated	✗	0.8	0.9604	0.9300	**1.0000**	0.8409
	✓	0.8	**0.9648**	**0.9764**	**1.0000**	**0.8591**
supplemental O2	✗	0.7	**0.9800**	0.9672	**1.0000**	**0.9000**
	✓	0.7	0.9609	**1.0000**	**1.0000**	0.8000

Comparison on supplemental O2 prediction Table 5.13 compares the performance of our method with benchmarks on supplemental oxygen prediction. The combination of Mean + ADASYN + LP showed the highest performance in terms of F1-score and our method showed the best performance in terms of all evaluation metrics except for F1-score. HexaGAN with hint rates 0, 0.7, 0.8 showed the AUROC of 1.0000, which indicates that our method can perfectly predict supplemental oxygen by adjusting the prediction threshold.

Effect of semi-supervised learning We conducted an ablation study on semi-supervised learning to investigate the effectiveness of utilizing other pneumonia data as unlabeled data. Table 5.14 demonstrates that treating records from other pneumonia as unlabeled data and employing semi-supervised learning improves prediction performances, particularly in terms of AUROC.

5.2.5 Discussion

Machine learning perspective

In order to learn a classifier from the training data in which the limited data problems exist simultaneously, preprocessing techniques to solve each problem must be applied sequentially. However, this approach requires building a preprocessing pipeline that depends on problems present in the training data. Because each technique focuses on a problem, it ignores the connections between the problems. However, HexaGAN has the advantage that there is no need to build a specific preprocessing pipeline. In addition, HexaGAN deals with limited data problems simultaneously through interaction between its components, which helps to maximize the prediction performance.

As mentioned in Chapter 3, the limited data problems can be solved via imputation. Existing ML methods to overcome the limited data problems either use the value of the nearby training data or perform only simple processing. This may result in the imputed data not faithfully following the data distribution. On the other hand, deep learning methods can learn complex and nonlinear patterns inherent in data. Our method estimates the data distribution using deep generative models and addresses the limited data problems more effectively. This eventually improves the prediction performance of mortality and interventions in COVID-19 patients.

Medical perspective

The model presented in this study can predict survival, the use of supplemental oxygen, and intubation which are closely related to the severity of COVID-19. Since the use of dexamethasone is important for the development and mortality of the disease [117], the effect of early use of dexamethasone according to the prediction of the model could be further investigated.

Since the dataset used for experiments was obtained from a cross-sectional study, the learned model can predict whether an intervention was observed in retrospective data. We expect that GAN methods trained on datasets obtained from longitudinal studies can predict whether an intervention should be recommended at a particular moment. This will be an important avenue for future work to provide more timely treatment to COVID-19 patients in the presence of the limited data problems.

5.3 Summary

In this chapter, we apply HexaGAN introduced in Chapter 3 to electronic health records with the imperfect dataset problem.

First, we developed a method for detecting the amyloid positivity of CN individuals using proxy measures including structural MRI scans, demographic information, and clinical scores with a deep generative model. In tandem with the growth of artificial intelligence (AI) systems for electronic health records (EHRs), deep generative models will effectively address the imperfect dataset problem in EHRs and allow us to successfully perform clinical translation. Moreover, the fusion of XAI techniques and statistical tests will help locate important regions and features for detecting diseases and provide the reliability of the model in the real-world.

Second, we have proposed a method using HexaGAN and a hint mechanism to predict mortality and the need for medical interventions. Our method robustly predicts mortality and interventions in the presence of the limited data problems such as missing data, missing label, and class imbalance problems. We also confirmed that a hint mechanism can enhance the prediction performance of HexaGAN. We believe that the excellent performance on mortality and intervention prediction of our model can help allocate limited medical resources efficiently, provide timely and appropriate clinical interventions, and ultimately save more lives from COVID-19.

Chapter 6

Conclusion

In this final chapter, we bring this dissertation to a close by providing a concise summary of the previous chapters. Additionally, we delve into comprehensive discussions on future research directions and present an outlook on the field of learning from limited data with deep generative models. By offering a forward-looking perspective, our aim is to inspire and guide future researchers in their endeavors to advance the field and unlock the full potential of deep generative models for learning from limited data.

6.1 Summary of Dissertation

In Chapter 3, the HexaGAN framework is introduced as a solution to address three main problems in real-world classification: missing data, class imbalance, and missing label. The HexaGAN framework incorporates six neural networks that actively correlate with each other, and novel loss functions are designed to maximize the utilization of incomplete data. The proposed method outperforms existing state-of-the-art methods in both imputation and classification tasks, providing a comprehensive solution to the three commonly encountered problems in real-world classification. Future work includes extending the HexaGAN framework to time series datasets, such as electronic health records.

Chapter 4 presents the SLOGAN method, which is designed for generating data conditioned on learned attributes in real-world datasets with balanced or imbalanced attribute distributions. The method utilizes implicit reparameterization and unsupervised conditional contrastive loss to achieve state-of-the-art unsupervised conditional generation performance. The incorporation of a small amount of probe data enables attribute control in the generated samples. Future research directions involve exploring principled methods to determine the number and hierarchy of attributes in real-world data and improving the quality of samples with minority attributes in unsupervised conditional GANs.

In Chapter 5, a method combining HexaGAN and a hint mechanism is proposed for mortality and medical intervention prediction. This method demonstrates robust performance in the presence of limited data problems, such as missing data, missing label, and class imbalance. The hint mechanism enhances the prediction performance of HexaGAN. The developed model holds the potential to efficiently allocate limited medical resources, provide timely and appropriate clinical interventions, and save lives in the context of COVID-19. Furthermore, the dissertation explores the application of deep generative models in detecting amyloid positivity in cognitively normal individuals using proxy measures, such as structural MRI scans, demographic information, and clinical scores. The utilization of deep generative models in electronic health records (EHRs) addresses the imperfect dataset problem and enables successful clinical translation. The fusion of Explainable AI (XAI) techniques and statistical tests enhances the interpretability and reliability of the model in real-world applications.

In summary, this dissertation presents novel methodologies and frameworks to address imperfect dataset problems in real-world classification, unsupervised conditional generation, and application to EHRs. The proposed solutions demonstrate superior performance compared to existing methods and hold great potential for applications in healthcare and biomedical data analysis.

6.2 Future Research and Outlook

6.2.1 Addressing new imperfect dataset problems

Beyond the imperfect dataset problems addressed in this dissertation, there are numerous opportunities to explore new challenges and domains in this field. The field of imperfect data is vast and constantly evolving, offering researchers exciting possibilities for exploration and innovation. By expanding the scope of the investigation, researchers can tackle novel problems related to imperfect datasets across various domains, including finance, social sciences, and environmental studies. These challenges may involve addressing issues like label noise and distribution shifts. Through rigorous research and innovative approaches, new insights and methodologies can be developed to effectively handle these imperfections.

In the field of AI healthcare, advancements can be made in accurately diagnosing medical conditions by handling imperfect datasets with DGMs, leading to improved patient care and treatment outcomes. Furthermore, this research can pave the way for the development of new medical AI products and services that effectively deal with imperfections and uncertainties in healthcare data. Machine vision applications, particularly in deep learning-based quality monitoring for smart factories, can also benefit from this research. By addressing imperfect dataset challenges, such as variations in lighting conditions or occlusions, DGMs can enhance the accuracy and reliability of quality assessment systems in industrial settings. This can lead to improved production processes and higher product quality.

The impact of this research can extend beyond specific domains that deal with imperfect training and test datasets. By effectively handling existing and new imperfect dataset problems, DGMs can enhance the performance and generalizability of deep learning models across diverse domains, including image recognition, natural language processing, and data classification.

6.2.2 DGMs for unsupervised conditional generation

Future work in unsupervised conditional generation (UCG) has great potential as researchers explore and enhance DGMs, including diffusion models. The objective is to develop and refine methodologies that effectively uncover salient attributes from unlabeled data and generate data with specific attributes. The outcomes of this research have wide-ranging and significant implications.

In the field of biomedical data science, researchers can unlock the ability to discover distinct subtypes from patient data. Subsequently, it becomes possible to generate data that represents these identified subtypes. This capability holds potential for various downstream tasks, including improved diagnosis, personalized treatment, and tailored interventions in healthcare. By leveraging UCG methods, researchers can make significant strides in improving healthcare outcomes and advancing medical knowledge. Moreover, this research also seeks to address new difficulties arising from the scarcity of patient data in unlabeled datasets. Robust UCG methods provide a practical solution by allowing for the augmentation of data, even in situations involving imperfect dataset problems.

6.2.3 Precision medicine

Resolving the challenges posed by imperfect datasets is crucial for advancing precision medicine. Precision medicine aims to provide personalized healthcare interventions based on individual characteristics like genetics, lifestyle, and environmental factors. By effectively handling imperfect datasets, researchers can extract valuable insights that enable the implementation of precise and tailored treatments.

Addressing imperfect dataset problems allows for the integration of various data sources, including genomics, proteomics, and clinical data. This comprehensive approach provides a holistic understanding of patients' conditions, leading to more accurate predictions of diagnoses and prognoses. By utilizing AI models trained on reliable and representative datasets, researchers can uncover complex relationships between patient attributes and treatment outcomes, paving the way for targeted therapies and interventions.

Moreover, tackling imperfect dataset problems empowers researchers to overcome issues related to data quality and bias. It ensures that AI models are trained and evaluated using trustworthy and unbiased data, thereby improving the reliability and fairness of the developed systems. This is essential to ensure that precision medicine benefits individuals from diverse backgrounds and populations.

In summary, actively addressing imperfect dataset problems in biomedical AI research not only enhances the reliability and accuracy of AI systems but also plays a vital role in advancing precision medicine. By harnessing the potential of comprehensive datasets and overcoming data imperfections, researchers can unlock the full potential of precision medicine, leading to improved patient outcomes, personalized treatments, and innovative changes in healthcare.

Bibliography

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 265–283, 2016.

[2] M Ehsan Abbasnejad, Anthony Dick, and Anton van den Hengel. Infinite variational autoencoder for semi-supervised learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 781–790. IEEE, 2017.

[3] Anees Abrol, Zening Fu, Mustafa Salman, Rogers Silva, Yuhui Du, Sergey Plis, and Vince Calhoun. Deep learning encodes robust discriminative neuroimaging representations to outperform standard machine learning. *Nature communications*, 12(1):1–17, 2021.

[4] Aida Ali, Siti Mariyam Shamsuddin, and Anca L Ralescu. Classification with class imbalance problem. *Int. J. Advance Soft Compu. Appl*, 5(3), 2013.

[5] K Abigail Andrews, Chris Frost, Marc Modat, M Jorge Cardoso, Chris C Rowe, Victor Villemagne, Nick C Fox, Sebastien Ourselin, and Jonathan M Schott. Acceleration of hippocampal atrophy rates in asymptomatic amyloidosis. *Neurobiology of aging*, 39:99–107, 2016.

[6] Yaseen M Arabi, Srinivas Murthy, and Steve Webb. Covid-19: a novel coronavirus and a novel challenge for critical care. *Intensive care medicine*, 46(5): 833–836, 2020.

[7] Martin Arjovsky and Leon Bottou. Towards principled methods for training generative adversarial networks. In *International Conference on Learning Representations*, 2017.

[8] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning*, pages 214–223, 2017.

[9] Mohammadreza Armandpour, Ali Sadeghian, Chunyuan Li, and Mingyuan Zhou. Partition-guided gans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5099–5109, 2021.

[10] Feng Bai, Zhijun Zhang, David R Watson, Hui Yu, Yongmei Shi, Wanlin Zhu, Liang Wang, Yonggui Yuan, and Yun Qian. Absent gender differences of hippocampal atrophy in amnestic type mild cognitive impairment. *Neuroscience letters*, 450(2):85–89, 2009.

[11] Mohammad M Banoei, Roshan Dinparastisaleh, Ali Vaeli Zadeh, and Mehdi Mirsaeidi. Machine-learning-based covid-19 mortality prediction model and identification of patients at low and high risk of dying. *Critical Care*, 25(1):1–14, 2021.

[12] Petronilla Battista, Christian Salvatore, and Isabella Castiglioni. Optimizing neuropsychological assessments for cognitive, behavioral, and functional impairment classification: a machine learning study. *Behavioural neurology*, 2017, 2017.

[13] Brett K Beaulieu-Jones, Casey S Greene, et al. Semi-supervised learning of

the electronic health record for phenotype stratification. *Journal of biomedical informatics*, 64:168–178, 2016.

[14] J Alex Becker, Trey Hedden, Jeremy Carmasin, Jacqueline Maye, Dorene M Rentz, Deepti Putcha, Bruce Fischl, Douglas N Greve, Gad A Marshall, Stephen Salloway, et al. Amyloid-β associated cortical thinning in clinically normal elderly. *Annals of neurology*, 69(6):1032–1042, 2011.

[15] Mohamed Ishmael Belghazi, Sai Rajeswar, Olivier Mastropietro, Negar Rostamzadeh, Jovana Mitrovic, and Aaron Courville. Hierarchical adversarially learned inference. *arXiv preprint arXiv:1802.01071*, 2018.

[16] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[17] Georges Bonnet. Transformations des signaux aléatoires a travers les systemes non linéaires sans mémoire. In *Annales des Télécommunications*, volume 19, pages 203–220. Springer, 1964.

[18] Taxiarchis Botsis, Gunnar Hartvigsen, Fei Chen, and Chunhua Weng. Secondary use of ehr: data quality issues and informatics opportunities. *Summit on Translational Bioinformatics*, 2010:1, 2010.

[19] Heiko Braak and Eva Braak. Neuropathological stageing of alzheimer-related changes. *Acta neuropathologica*, 82(4):239–259, 1991.

[20] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[21] S van Buuren and Karin Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in r. *Journal of statistical software*, pages 1–68, 2010.

[22] Adrià Casamitjana, Paula Petrone, Alan Tucholka, Carles Falcon, Stavros Skouras, José Luis Molinuevo, Verónica Vilaplana, Juan Domingo Gispert, Alzheimer's Disease Neuroimaging Initiative, et al. Mri-based screening of preclinical alzheimer's disease for prevention clinical trials. *Journal of Alzheimer's Disease*, 64(4):1099–1112, 2018.

[23] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15, 2009.

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

[25] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. Recurrent neural networks for multivariate time series with missing values. *Scientific reports*, 8(1):6085, 2018.

[26] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[27] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: interpretable representation learning by information maximizing generative adversarial nets. In *Neural Information Processing Systems (NIPS)*, 2016.

[28] Soo Hyun Cho, Yeong Sim Choe, Young Ju Kim, Hee Jin Kim, Hyemin Jang, Yeshin Kim, Si Eun Kim, Seung Joo Kim, Jun Pyo Kim, Young Hee Jung, et al. Head-to-head comparison of 18f-florbetaben and 18f-flutemetamol in the cortical and striatal regions. *Journal of Alzheimer's Disease*, (Preprint):1–10, 2020.

[29] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[30] Joseph Paul Cohen, Paul Morrison, and Lan Dao. Covid-19 image data collection. *arXiv 2003.11597*, 2020. URL https://github.com/ieee8023/covid-chestxray-dataset.

[31] Marta Crous-Bou, Carolina Minguillón, Nina Gramunt, and José Luis Molinuevo. Alzheimer's disease prevention: from risk factors to early intervention. *Alzheimer's research & therapy*, 9(1):71, 2017.

[32] Jeffrey Cummings, Garam Lee, Aaron Ritter, Marwan Sabbagh, and Kate Zhong. Alzheimer's disease drug development pipeline: 2019. *Alzheimer's & Dementia: Translational Research & Clinical Interventions*, 5:272–293, 2019.

[33] Zihang Dai, Zhilin Yang, Fan Yang, William W Cohen, and Ruslan R Salakhutdinov. Good semi-supervised learning that requires a bad gan. In *Advances in Neural Information Processing Systems*, pages 6510–6520, 2017.

[34] Bart Marius de Vries, Sandeep SV Golla, Jarith Ebenau, Sander CJ Verfaillie, Tessa Timmers, Fiona Heeman, Matthijs CF Cysouw, Bart NM van Berckel, Wiesje M van der Flier, Maqsood Yaqub, et al. Classification of negative and positive 18 f-florbetapir brain pet studies in subjective cognitive decline patients using a convolutional neural network. *European Journal of Nuclear Medicine and Molecular Imaging*, 48(3):721–728, 2021.

[35] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2019.

[36] Dua Dheeru and Efi Karra Taniskidou. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

[37] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. In *5th International Conference on Learning Representations*, 2017.

[38] Qi Dong, Shaogang Gong, and Xiatian Zhu. Class rectification hard mining for imbalanced deep learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1851–1860, 2017.

[39] Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Alex Lamb, Martín Arjovsky, Olivier Mastropietro, and Aaron C. Courville. Adversarially learned inference. In *5th International Conference on Learning Representations*, 2017.

[40] AbdenNaji El Fadly, Bastien Rance, Noël Lucas, Charles Mead, Gilles Chatellier, Pierre-Yves Lastic, Marie-Christine Jaulent, and Christel Daniel. Integrating clinical research with the healthcare enterprise: from the re-use project to the ehr4cr platform. *Journal of biomedical informatics*, 44:S94–S102, 2011.

[41] Shaza M Abd Elrahman and Ajith Abraham. A review of class imbalance problem. *Journal of Network and Innovative Computing*, 1(2013):332–340, 2013.

[42] Shaza M Abd Elrahman and Ajith Abraham. A review of class imbalance problem. *Journal of Network and Innovative Computing*, 1(2013):332–340, 2013.

[43] Justin Engelmann and Stefan Lessmann. Conditional wasserstein gan-based oversampling of tabular data for imbalanced learning. *Expert Systems with Applications*, 174:114582, 2021.

[44] Gill Farrar. Regional visual read inspection of [18f] flutemetamol brain images

from end-of-life and amnestic mci subjects. *Journal of Nuclear Medicine*, 58 (supplement 1):1250–1250, 2017.

[45] Mahyar Fazlyab, Alexander Robey, Hamed Hassani, Manfred Morari, and George Pappas. Efficient and accurate estimation of lipschitz constants for deep neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

[46] Mikhail Figurnov, Shakir Mohamed, and Andriy Mnih. Implicit reparameterization gradients. *Advances in Neural Information Processing Systems*, 31, 2018.

[47] Jerome H Friedman. On bias, variance, 0/1—loss, and the curse-of-dimensionality. *Data mining and knowledge discovery*, 1(1):55–77, 1997.

[48] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016.

[49] Erik Gawehn, Jan A Hiss, and Gisbert Schneider. Deep learning in drug discovery. *Molecular informatics*, 35(1):3–14, 2016.

[50] Sang gil Lee, Wei Ping, Boris Ginsburg, Bryan Catanzaro, and Sungroh Yoon. BigVGAN: A universal neural vocoder with large-scale training. In *The Eleventh International Conference on Learning Representations*, 2023.

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

[52] Ioannis S Gousias, Daniel Rueckert, Rolf A Heckemann, Leigh E Dyet, James P Boardman, A David Edwards, and Alexander Hammers. Automatic segmentation

of brain mris of 2-year-olds into 83 regions of interest. *Neuroimage*, 40(2):672–684, 2008.

[53] Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. In *Advances in neural information processing systems*, pages 529–536, 2005.

[54] Alex Graves. Stochastic backpropagation through mixture density distributions. *arXiv preprint arXiv:1607.05690*, 2016.

[55] Michel J Grothe, Henryk Barthel, Jorge Sepulcre, Martin Dyrba, Osama Sabri, Stefan J Teipel, Alzheimer's Disease Neuroimaging Initiative, et al. In vivo staging of regional amyloid deposition. *Neurology*, 89(20):2031–2038, 2017.

[56] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pages 5767–5777, 2017.

[57] Tracy D Gunter and Nicolas P Terry. The emergence of national electronic health record architectures in the united states and australia: models, costs, and questions. *Journal of medical Internet research*, 7(1), 2005.

[58] Pavel Gurevich, Hannes Stuke, Andreas Kastrup, Heiner Stuke, and Helmut Hildebrandt. Neuropsychological testing and machine learning distinguish alzheimer's disease from other causes for cognitive impairment. *Frontiers in aging neuroscience*, 9:114, 2017.

[59] Swaminathan Gurumurthy, Ravi Kiran Sarvadevabhatla, and R Venkatesh Babu. Deligan: Generative adversarial networks for diverse and limited data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 166–174, 2017.

[60] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

[61] Alice Hahn, Young Ju Kim, Hee Jin Kim, Hyemin Jang, Hanna Cho, Seong Hye Choi, Byeong C Kim, Kyung Won Park, Duk L Na, Juhee Chin, et al. the preclinical amyloid sensitive composite to determine subtle cognitive differences in preclinical alzheimer's disease. *Scientific Reports*, 10(1):1–11, 2020.

[62] Alexander Hammers, Richard Allom, Matthias J Koepp, Samantha L Free, Ralph Myers, Louis Lemieux, Tejal N Mitchell, David J Brooks, and John S Duncan. Three-dimensional maximum probability atlas of the human brain, with particular reference to the temporal lobe. *Human brain mapping*, 19(4):224–247, 2003.

[63] Trevor Hastie, Rahul Mazumder, Jason D Lee, and Reza Zadeh. Matrix completion and low-rank svd via fast alternating least squares. *The Journal of Machine Learning Research*, 16(1):3367–3402, 2015.

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

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

[66] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6629–6640, 2017.

[67] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[68] Uiwon Hwang, Sungwoon Choi, Han-Byoel Lee, and Sungroh Yoon. Adversarial training for disease prediction from electronic health records with missing data. *arXiv preprint arXiv:1711.04126*, 2017.

[69] Uiwon Hwang, Sungwoon Choi, and Sungroh Yoon. Disease prediction from electronic health records using generative adversarial networks. *arXiv preprint arXiv:1711.04126*, 2017.

[70] Uiwon Hwang, Dahuin Jung, and Sungroh Yoon. Hexagan: Generative adversarial nets for real world classification. In *International Conference on Machine Learning*, pages 2921–2930. PMLR, 2019.

[71] Leonardo Iaccarino, Gautam Tammewar, Nagehan Ayakta, Suzanne L Baker, Alexandre Bejanin, Adam L Boxer, Maria Luisa Gorno-Tempini, Mustafa Janabi, Joel H Kramer, Andreas Lazaris, et al. Local and distant relationships between amyloid, tau and neurodegeneration in alzheimer's disease. *NeuroImage: Clinical*, 17:452–464, 2018.

[72] Ayush Jaiswal, Wael AbdAlmageed, Yue Wu, and Premkumar Natarajan. Bidirectional conditional generative adversarial networks. In *Asian Conference on Computer Vision*, pages 216–232. Springer, 2018.

[73] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017.

[74] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-

tailed recognition. In *International Conference on Learning Representations*, 2020.

[75] Bingyi Kang, Yu Li, Sa Xie, Zehuan Yuan, and Jiashi Feng. Exploring balanced feature spaces for representation learning. In *International Conference on Learning Representations*, 2021.

[76] Minguk Kang and Jaesik Park. Contragan: Contrastive learning for conditional image generation. In *Advances in Neural Information Processing Systems*, 2020.

[77] Andrei Kapishnikov, Tolga Bolukbasi, Fernanda Viégas, and Michael Terry. Xrai: Better attributions through regions. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4948–4957, 2019.

[78] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

[79] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

[80] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8110–8119, 2020.

[81] Mohammed Khalilia, Sounak Chakraborty, and Mihail Popescu. Predicting disease risks from highly imbalanced data using random forest. *BMC medical informatics and decision making*, 11(1):51, 2011.

[82] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[83] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[84] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems*, 31, 2018.

[85] Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*, pages 3581–3589, 2014.

[86] Hyunwoong Ko, Jung-Joon Ihm, Hong-Gee Kim, Alzheimer's Disease Neuroimaging Initiative, et al. Cognitive profiling related to cerebral amyloid beta burden using machine learning approaches. *Frontiers in aging neuroscience*, 11, 2019.

[87] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, (8):30–37, 2009.

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

[89] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *International Conference on Learning Representations*, 2017.

[90] Susan M Landau, Christopher Breault, Abhinay D Joshi, Michael Pontecorvo, Chester A Mathis, William J Jagust, and Mark A Mintun. Amyloid-β imaging with pittsburgh compound b and florbetapir: comparing radiotracers and quantification methods. *Journal of Nuclear Medicine*, 54(1):70–77, 2013.

[91] Susan M Landau, Andy Horng, Allison Fero, William J Jagust, Alzheimer's Disease Neuroimaging Initiative, et al. Amyloid negativity in patients with clinically diagnosed alzheimer disease and mci. *Neurology*, 86(15):1377–1385, 2016.

[92] O Langford, R Raman, RA Sperling, J Cummings, C-K Sun, G Jimenez-Maggiora, PS Aisen, and MC Donohue. Predicting amyloid burden to accelerate recruitment of secondary prevention clinical trials. *The journal of prevention of Alzheimer's disease*, 7(4):213–218, 2020.

[93] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11): 2278–2324, 1998.

[94] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521 (7553):436–444, 2015.

[95] Chongxuan Li, Taufik Xu, Jun Zhu, and Bo Zhang. Triple generative adversarial nets. In *Advances in Neural Information Processing Systems 30*, pages 4088–4098. 2017.

[96] Rena Li and Meharvan Singh. Sex differences in cognitive impairment and alzheimer's disease. *Frontiers in neuroendocrinology*, 35(3):385–403, 2014.

[97] Wu Lin, Mohammad Emtiyaz Khan, and Mark Schmidt. Fast and simple natural-gradient variational inference with mixture of exponential-family approximations. In *International Conference on Machine Learning*, pages 3992–4002. PMLR, 2019.

[98] Wu Lin, Mohammad Emtiyaz Khan, and Mark Schmidt. Stein's lemma for the reparameterization trick with exponential family mixtures. *arXiv preprint arXiv:1910.13398*, 2019.

[99] Wu Lin, Mark Schmidt, and Mohammad Emtiyaz Khan. Handling the positive-definite constraint in the Bayesian learning rule. In *Proceedings of the 37th International Conference on Machine Learning*, pages 6116–6126, 2020.

[100] Steven Liu, Tongzhou Wang, David Bau, Jun-Yan Zhu, and Antonio Torralba. Diverse image generation via self-conditioned gans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14286–14295, 2020.

[101] 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.

[102] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

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

[104] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In *International Conference on Machine Learning*, pages 3478–3487, 2018.

[105] Michelle M Mielke, Prashanthi Vemuri, and Walter A Rocca. Clinical epidemiology of alzheimer's disease: assessing sex and gender differences. *Clinical epidemiology*, 6:37, 2014.

[106] Seonwoo Min, Byunghan Lee, and Sungroh Yoon. Deep learning in bioinformatics. *Briefings in bioinformatics*, 18(5):851–869, 2017.

[107] Seonwoo Min, Byunghan Lee, and Sungroh Yoon. Deep learning in bioinformatics. *Briefings in bioinformatics*, 18(5):851–869, 2017.

[108] Riccardo Miotto, Li Li, Brian A Kidd, and Joel T Dudley. Deep patient: An unsupervised representation to predict the future of patients from the electronic health records. *Scientific reports*, 6:26094, 2016.

[109] Riccardo Miotto, Li Li, Brian A Kidd, and Joel T Dudley. Deep patient: an unsupervised representation to predict the future of patients from the electronic health records. *Scientific reports*, 6:26094, 2016.

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

[111] Takeru Miyato and Masanori Koyama. cgans with projection discriminator. In *International Conference on Learning Representations*, 2018.

[112] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, Ken Nakae, and Shin Ishii. Distributional smoothing with virtual adversarial training. *arXiv preprint arXiv:1507.00677*, 2015.

[113] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=B1QRgziT-.

[114] Shakir Mohamed and Balaji Lakshminarayanan. Learning in implicit generative models. *arXiv preprint arXiv:1610.03483*, 2016.

[115] Elizabeth C Mormino, Rebecca A Betensky, Trey Hedden, Aaron P Schultz, Rebecca E Amariglio, Dorene M Rentz, Keith A Johnson, and Reisa A Sperling. Synergistic effect of β-amyloid and neurodegeneration on cognitive decline in clinically normal individuals. *JAMA neurology*, 71(11):1379–1385, 2014.

[116] Sudipto Mukherjee, Himanshu Asnani, Eugene Lin, and Sreeram Kannan. Clustergan: Latent space clustering in generative adversarial networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 4610–4617, 2019.

[117] NIH. Covid-19 treatment guidelines. https://www.covid19treatmentguidelines.nih.gov/management/clinical-management/clinical-management-summary/, 2022. [Online; accessed 31-May-2022].

[118] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *International conference on machine learning*, pages 2642–2651. PMLR, 2017.

[119] Sebastian Palmqvist, Michael Schöll, Olof Strandberg, Niklas Mattsson, Erik Stomrud, Henrik Zetterberg, Kaj Blennow, Susan Landau, William Jagust, and Oskar Hansson. Earliest accumulation of β-amyloid occurs within the default-mode network and concurrently affects brain connectivity. *Nature communications*, 8(1):1–13, 2017.

[120] Lili Pan, Peijun Tang, Zhiyong Chen, and Zenglin Xu. Contrastive disentanglement in generative adversarial networks. *arXiv preprint arXiv:2103.03636*, 2021.

[121] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[122] Jae Young Park, Sungroh Yoon, Man Sik Park, Dae-Yeon Cho, Hong-Seok Park, Du Geon Moon, and Duck Ki Yoon. Initial biopsy outcome prediction in korean patients-comparison of a noble web-based korean prostate cancer risk calculator versus prostate-specific antigen testing. *Journal of Korean medical science*, 26 (1):85–91, 2011.

[123] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[124] Timo Pekkala, Anette Hall, Tiia Ngandu, Mark van Gils, Seppo Helisalmi, Tuomo Hänninen, Nina Kemppainen, Yawu Liu, Jyrki Lötjönen, Teemu Paajanen, et al. Detecting amyloid positivity in elderly with increased risk of cognitive decline. *Frontiers in Aging Neuroscience*, 12, 2020.

[125] Paula M Petrone, Adrià Casamitjana, Carles Falcon, Miquel Artigues, Grégory Operto, Raffaele Cacciaglia, José Luis Molinuevo, Verónica Vilaplana, Juan Domingo Gispert, Alzheimer's Disease Neuroimaging Initiative, et al. Prediction of amyloid pathology in cognitively unimpaired individuals using voxel-wise analysis of longitudinal structural brain mri. *Alzheimer's research & therapy*, 11(1):72, 2019.

[126] Henning Petzka, Asja Fischer, and Denis Lukovnikov. On the regularization of wasserstein GANs. In *International Conference on Learning Representations*, 2018.

[127] Ben Poole, Sherjil Ozair, Aaron Van Den Oord, Alex Alemi, and George Tucker. On variational bounds of mutual information. In *International Conference on Machine Learning*, pages 5171–5180. PMLR, 2019.

[128] Robert Price. A useful theorem for nonlinear devices having gaussian inputs. *IRE Transactions on Information Theory*, 4(2):69–72, 1958.

[129] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *International Conference on Learning Representations*, 2016.

[130] William M Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, 66(336):846–850, 1971.

[131] Laila Rasmy, Yang Xiang, Ziqian Xie, Cui Tao, and Degui Zhi. Med-bert: pretrained contextualized embeddings on large-scale structured electronic health records for disease prediction. *NPJ digital medicine*, 4(1):1–13, 2021.

[132] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[133] Douglas A Reynolds et al. Gaussian mixture models. *Encyclopedia of biometrics*, 741(659-663), 2009.

[134] Shannon L Risacher, Sungeun Kim, Li Shen, Kwangsik Nho, Tatiana Foroud, Robert C Green, Ronald C Petersen, Clifford R Jack Jr, Paul S Aisen, Robert A Koeppe, et al. The role of apolipoprotein e (apoe) genotype in early mild cognitive impairment (e-mci). *Frontiers in aging neuroscience*, 5:11, 2013.

[135] Donald B Rubin. Inference and missing data. *Biometrika*, 63(3):581–592, 1976.

[136] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*, pages 2234–2242, 2016.

[137] Saranya Sankaranarayanan, Jagadheshwar Balan, Jesse R Walsh, Yanhong Wu, Sara Minnich, Amy Piazza, Collin Osborne, Gavin R Oliver, Jessica Lesko, Kathy L Bates, et al. Covid-19 mortality prediction from deep learning in a large multistate electronic health record and laboratory information system data set: Algorithm development and validation. *Journal of medical Internet research*, 23 (9):e30157, 2021.

[138] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? *Advances in neural information processing systems*, 31, 2018.

[139] Chao Shang, Aaron Palmer, Jiangwen Sun, Ko-Shin Chen, Jin Lu, and Jinbo Bi. Vigan: Missing view imputation with generative adversarial networks. In *Big Data (Big Data), 2017 IEEE International Conference on*, pages 766–775. IEEE, 2017.

[140] Konstantin Shmelkov, Cordelia Schmid, and Karteek Alahari. How good is my gan? In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 213–229, 2018.

[141] Jost Tobias Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. *arXiv preprint arXiv:1511.06390*, 2015.

[142] Jean Stawiaski. A pretrained densenet encoder for brain tumor segmentation. In *International MICCAI Brainlesion Workshop*, pages 105–115. Springer, 2018.

[143] Charles M Stein. Estimation of the mean of a multivariate normal distribution. *The annals of Statistics*, pages 1135–1151, 1981.

[144] Yanmin Sun, Mohamed S Kamel, Andrew KC Wong, and Yang Wang. Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition*, 40 (12):3358–3378, 2007.

[145] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 3319–3328. JMLR.org, 2017.

[146] Qiuling Suo, Fenglong Ma, Ye Yuan, Mengdi Huai, Weida Zhong, Aidong Zhang, and Jing Gao. Personalized disease prediction using a cnn-based similarity learning method. In *Bioinformatics and Biomedicine (BIBM), 2017 IEEE International Conference on*, 2017.

[147] Harini Suresh, Nathan Hunt, Alistair Johnson, Leo Anthony Celi, Peter Szolovits, and Marzyeh Ghassemi. Clinical intervention prediction and understanding with deep neural networks. In *Machine Learning for Healthcare Conference*, pages 322–337, 2017.

[148] Ilya Sutskever, Rafal Jozefowicz, Karol Gregor, Danilo Rezende, Tim Lillicrap, and Oriol Vinyals. Towards principled unsupervised learning. *arXiv preprint arXiv:1511.06440*, 2015.

[149] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[150] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in neural information processing systems*, pages 1195–1204, 2017.

[151] Mara Ten Kate, Alberto Redolfi, Enrico Peira, Isabelle Bos, Stephanie J Vos, Rik Vandenberghe, Silvy Gabel, Jolien Schaeverbeke, Philip Scheltens, Olivier Blin, et al. Mri predictors of amyloid pathology: results from the emif-ad multimodal biomarker discovery study. *Alzheimer's research & therapy*, 10(1):1–12, 2018.

[152] Dávid Terjék. Adversarial lipschitz regularization. In *International Conference on Learning Representations*, 2020.

[153] L Theis, A van den Oord, and M Bethge. A note on the evaluation of generative models. In *International Conference on Learning Representations (ICLR 2016)*, pages 1–10, 2016.

[154] Kelsey R Thomas, Joel Eppig, Emily C Edmonds, Diane M Jacobs, David J Libon, Rhoda Au, David P Salmon, and Mark W Bondi. Word-list intrusion errors predict progression to mild cognitive impairment. *Neuropsychology*, 32 (2):235, 2018.

[155] Tijmen Tieleman and Geoffrey Hinton. Rmsprop: Divide the gradient by a running average of its recent magnitude. *Coursera: Neural networks for machine learning*, 4:26–31, 2012.

[156] Duygu Tosun, Dallas Veitch, Paul Aisen, Clifford R Jack Jr, William J Jagust, Ronald C Petersen, Andrew J Saykin, James Bollinger, Vitaliy Ovod, Kwasi G Mawuenyega, et al. Detection of β-amyloid positivity in alzheimer's disease neuroimaging initiative participants with demographics, cognition, mri and plasma biomarkers. *Brain communications*, 3(2):fcab008, 2021.

[157] Olga Troyanskaya, Michael Cantor, Gavin Sherlock, Pat Brown, Trevor Hastie, Robert Tibshirani, David Botstein, and Russ B Altman. Missing value estimation methods for dna microarrays. *Bioinformatics*, 17(6):520–525, 2001.

[158] Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 384–394. Association for Computational Linguistics, 2010.

[159] Stef Van Buuren. *Flexible imputation of missing data*. Chapman and Hall/CRC, 2018.

[160] Vladimir Vapnik. Principles of risk minimization for learning theory. *Advances in neural information processing systems*, 4, 1991.

[161] Zahra Asghari Varzaneh, Azam Orooji, Leila Erfannia, and Mostafa Shanbehzadeh. A new covid-19 intubation prediction strategy using an intelligent feature selection and k-nn method. *Informatics in Medicine Unlocked*, 28:100825, 2022.

[162] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103. ACM, 2008.

[163] Fei Wang and Changshui Zhang. Label propagation through linear neighborhoods. *IEEE Transactions on Knowledge and Data Engineering*, 20(1):55–67, 2008.

[164] Yaxing Wang, Chenshen Wu, Luis Herranz, Joost van de Weijer, Abel Gonzalez-Garcia, and Bogdan Raducanu. Transferring gans: generating images from limited data. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 218–234, 2018.

[165] Zifan Wang, Haofan Wang, Shakul Ramkumar, Piotr Mardziel, Matt Fredrikson, and Anupam Datta. Smoothed geometry for robust attribution. In *Advances in Neural Information Processing Systems*, 2020.

[166] Tingyi Wanyan, Jing Zhang, Ying Ding, Ariful Azad, Zhangyang Wang, and Benjamin S Glicksberg. Bootstrapping your own positive sample: Contrastive learning with electronic health record data. *arXiv preprint arXiv:2104.02932*, 2021.

[167] Jeremy C Weiss. *Statistical Timeline Analysis for Electronic Health Records*. PhD thesis, The University of Wisconsin-Madison, 2014.

[168] Jenna Wiens, John Guttag, and Eric Horvitz. A study in transfer learning: leveraging data from multiple hospitals to enhance hospital-specific predictions. *Journal of the American Medical Informatics Association*, 21(4):699–706, 2014.

[169] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[170] Huan Xu and Shie Mannor. Robustness and generalization. *Machine learning*, 86(3):391–423, 2012.

[171] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. Gain: Missing data imputation using generative adversarial nets. In *Proceedings of the 35th International Conference on Machine Learning*, pages 5689–5698. PMLR, 2018.

[172] William J Youden. Index for rating diagnostic tests. *Cancer*, 3(1):32–35, 1950.

[173] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

[174] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657, 2015.

[175] Shengyu Zhao, Zhijian Liu, Ji Lin, Jun-Yan Zhu, and Song Han. Differentiable augmentation for data-efficient gan training. *Advances in Neural Information Processing Systems*, 33, 2020.

[176] Grace XY Zheng, Jessica M Terry, Phillip Belgrader, Paul Ryvkin, Zachary W Bent, Ryan Wilson, Solongo B Ziraldo, Tobias D Wheeler, Geoff P McDermott, Junjie Zhu, et al. Massively parallel digital transcriptional profiling of single cells. *Nature communications*, 8(1):1–12, 2017.

[177] Huasong Zhong, Chong Chen, Zhongming Jin, and Xian-Sheng Hua. Deep robust clustering by contrastive learning. *arXiv preprint arXiv:2008.03030*, 2020.

[178] Huasong Zhong, Chong Chen, Zhongming Jin, and Xian-Sheng Hua. Deep robust clustering by contrastive learning. *arXiv preprint arXiv:2008.03030*, 2020.

[179] Xiaojin Zhu and Zoubin Ghahramani. Learning from labeled and unlabeled data with label propagation. 2002.

[180] Xiaojin Zhu and Zoubin Ghahramaniн. Learning from labeled and unlabeled data with label propagation. 2002.

초 록

딥러닝은 데이터로부터 유용한 인사이트를 자동으로 추출하는 데에 있어서 주목할 만한 발전을 이루어왔다. 그러나 실세계에서는 종종 학습 데이터가 제한적이며, 결측값, 클래스 및 속성 불균형, 라벨 결측 등의 문제가 발생할 수 있다. 이러한 문제들은 실세계 데이터와 딥러닝 모델의 가정 사이에 불일치를 초래하며, 도메인 지식과 인력을 필요로 하는 전처리 단계를 요구한다. 따라서, 제한된 데이터를 효과적으로 학습할 수 있는 강건한 인공지능 시스템을 개발하는 것이 중요하다.

이러한 문제들에 대응하기 위해, 우리는 딥 생성 모델 (DGM)을 활용하였다. DGM은 딥러닝을 통해 데이터 분포를 추정하는 기법으로 최근 상당한 발전을 이루었다. 본 논문은 DGM의 일종인 생성적 적대 신경망 (GAN)을 중심으로 제한된 데이터셋과 관련된 문제들을 해결하기 위한 새로운 방법을 고안하고 응용한다. 본 논문의 주요 연구 주제는 실세계 분류를 위한 GAN, 비지도 조건부 생성을 위한 GAN, 의생명 데이터를 위한 DGM 응용이다.

첫 번째 연구 주제는 실세계 분류이며, 결측값, 클래스 불균형 및 레이블 결측과 같은 문제들을 효과적으로 처리할 수 있는 강건한 분류기를 학습하는 것을 목표로 한다. 이전 연구에서는 분류기를 훈련하기 전에 각각의 문제를 해결하기 위한 머신러닝 기반의 전처리 방법을 적용하였다. 하지만, 우리는 이 문제들을 “imputation”이라는 하나의 키워드로 재정의하고, 문제들 간의 상호 연관성을 고려한 새로운 GAN 기반의 프레임워크인 HexaGAN을 제안한다.

두 번째 연구 주제에서는 라벨이 없는 데이터로부터 조건부 생성을 수행하는 비지도 조건부 생성 (UCG)에 집중한다. DGM의 발전에도 불구하고, 조건부 생성은 여전히 대량의 라벨링된 데이터를 필요로 한다. 하지만, 실세계 데이터셋에는 라벨이 없는 경우가 흔하다. 이 문제를 해결하기 위해 데이터의 중요한 속성을 식별하고, 해당 속성을 포함하는 데이터를 생성하는 UCG 방법이 제안되었다. 그러나 기존의

UCG 모델은 속성이 균형적으로 분포되어 있다고 가정하여 불균형한 속성 학습에 실패하는 문제가 있다. 이를 극복하기 위해, 우리는 불균형한 속성을 강건하게 학습할 수 있는 SLOGAN을 제안한다.

마지막 연구 주제에서는 DGM을 의생명 데이터에 적용하여 결측 데이터, 클래스 불균형 및 레이블 부재와 같은 문제들을 해결한다. 첫 번째로, 우리는 침습적인 측정 대신 MRI 스캔, 인구통계학적 변수 및 인지 점수와 같은 측정을 통해 전임상 알츠하이머병을 예측하는 DGM을 제안한다. 이 접근법은 저렴하고 비침습적인 진단을 가능하게 하는 동시에, 병원 간 전이 가능성과 해석 가능성을 포함하는 딥러닝 모델의 임상적 응용에 필요한 요구 사항을 충족시킨다. 두 번째로, 우리는 HexaGAN과 힌트 메커니즘을 사용하여 COVID-19 환자의 생존과 임상적 개입을 예측한다. 이러한 방법은 한정된 데이터 문제에 대한 기존 기법의 조합보다 우수한 성능을 보여준다.

본 논문은 DGM의 잠재력을 활용하여 실세계 시나리오에서의 제한된 데이터 문제들을 해결함으로써 딥러닝 모델과 실제 응용 분야 사이의 간극을 좁히고자 한다. 또한, 다양한 분야의 제한된 데이터로부터 학습하는 미래 연구에 통찰력을 제공하기를 기대한다.

주요어: 제한된 데이터로부터의 학습, 딥 생성 모델, 딥러닝, 의생명 데이터 과학

학번: 2017-25277

감사의 글

학위 과정을 돌아보면 그간 많은 분들의 도움 덕분에 무사히 마무리할 수 있었습니다. 먼저 부족한 저를 연구실의 일원으로 받아주시고 독립적인 연구자로 성장할 수 있도록 아낌없는 지원과 세심한 지도를 해주신 윤성로 교수님께 진심으로 감사드립니다. 학부 과정 동안 머신러닝의 세계를 소개해 주신 성준경 교수님께 감사인사를 드립니다. 바쁜 일정 가운데 학위논문을 심사해 주시고 개선에 많은 도움을 주신 원중호 교수님, 이종환 교수님, 문태섭 교수님, 오민환 교수님께 감사드립니다.

연구의 즐거움을 함께 나눌 수 있었던 연구실 선후배들과 동료들에게 깊은 감사의 말씀을 전합니다. 연구에 있어 모범을 보여주신 호형, 헌석이형, 상길이형, 용준이형, 그리고 모든 연구실 선배님들께 감사드립니다. 열정적인 ML팀 동료들인 종현, 시원, 지수, 상하, 주현, 영탁, 세형이형, 논문의 공저자로 많은 도움을 주신 다휜누나, 희승, 현규형, 혜미, 그리고 모든 연구실 구성원분들의 도움으로 이 학위논문을 완성할 수 있었습니다. 그리고 공동연구를 함께한 성우형, 승욱, 재우, 의득, 민수씨에게도 감사드립니다. 모두들 원하시는 일 이루시고 행복하시기를 바랍니다.

언제나 제 행복을 위해 기도해 주시고, 따뜻한 격려와 응원을 보내주신 아버지, 어머니, 그리고 동생들에게 감사의 마음을 전하며 사랑한다고 말씀드리고 싶습니다. 긴 학위 과정 동안 가족들의 한결같은 지지가 힘든 순간들을 견딜 수 있었던 원동력이었습니다.

마지막으로, 학위 과정을 마치고 학자로서 드넓은 세상을 향해 날개를 펼치고자 합니다. 오늘의 제가 있기까지 지지와 격려를 보내주신 모든 분들께 다시 한번 감사의 말씀을 드립니다.

2023년 8월

황 의 원