Synthetic Electronic Medical Record Generation using Generative Adversarial Networks

Mohammadreza Beyki

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Science and Applications

Edward A. Fox, Chair
Jia-Bin Huang
Hoda Eldardiry

May 10, 2021
Blacksburg, Virginia

Keywords: Deep Learning, Healthcare, Generative Adversarial Networks

Synthetic Electronic Medical Record Generation using Generative Adversarial Networks

Mohammadreza Beyki

(ABSTRACT)

It has been a while that computers have replaced our record books, and medical records are no exception. Electronic Health Records (EHRs) are digital versions of a patient's medical records. EHRs are available to authorized users, and they should help doctors understand a patient's condition quickly. In recent years, Deep Learning models have proved their value and have become state-of-the-art in computer vision, natural language processing, speech, and other areas. The private nature of EHR data has prevented public access to EHR datasets. There are many obstacles to create a deep learning model with EHR data. Because EHR data primarily consists of huge sparse matrices, these challenges are mostly unique to this field. Due to this, research in this area is limited, and we can improve existing research substantially. In this study, we focus on high-performance synthetic data generation in EHR datasets. Artificial data generation can help reduce privacy leakage for dataset owners as there are research articles that describe re-identification attacks that undo de-identification methods. We propose a novel approach we call Improved Correlation Capturing Wasserstein Generative Adversarial Network (SCorGAN) to create EHR data. This work leverages Deep Convolutional Neural Networks to extract and understand spatial dependencies in EHR data. To improve our model's performance, we focus on our Deep Convolutional Autoencoder to better map our real EHR data to our latent space where we train the Generator. To assess our model's performance, we demonstrate that our generative model can create excellent data statistically close to the input dataset. Additionally, we evaluate our synthetic

dataset against the original data using our previous work that focused on GAN Performance Evaluation. This work is publicly available at https://github.com/mohibeyki/SCorGAN.

Synthetic Electronic Medical Record Generation using Generative Adversarial Networks

Mohammadreza Beyki

(GENERAL AUDIENCE ABSTRACT)

Artificial Intelligence (AI) systems have improved greatly in recent years. They are being used to understand all kinds of data. A practical use case for AI systems is to leverage their power to identify illnesses and find correlations between different conditions. To train AI and Machine Learning systems, we need to feed them huge datasets, and in the training process, we need to guide them so that they learn different features in our data. The more data an intelligent system has seen, the better it performs. However, health records are private, and we cannot share people's health records with the public, even for research purposes. This study provides a novel approach to synthetic data generation that others can use with intelligent systems. We show that our synthetic dataset is a good substitute for real datasets to train intelligent systems. Then these systems can work with actual health records and give accurate feedback on people's health conditions. Lastly, we present an intelligent system that we have trained using synthetic datasets to identify illnesses in a real dataset, with high accuracy and precision.

Dedication

To my wife Morva, whose unyielding love and support has given me strength to pursue and complete this study. To my parents who supported me all through my life to achieve my goals.

Acknowledgments

Throughout my graduate studies, I had the pleasure of working with fantastic people at Virginia Tech who supported me. First, I want to thank my supervisor, Professor Edward A. Fox. He taught me to think like a scientist and act like one. He has inspired me always to be the best person I can be. I would also like to acknowledge my brilliant labmates from the Digital Library Research Laboratory for their support. I would particularly like to single out my collaborator Dr. Amirsina Torfi. I want to thank him for his teaching, comfort, and patience to help me advance my research.

Contents

List of Figures

List of Tables

List of Abbreviations

AE	Autoencoder
AI	Artificial Intelligence
AIS	Annealed Importance Sampling
AUC	Area Under Curve
BCE	Binary Cross Entropy
CAE	Convolutional Autoencoder
CNN	Convolutional Neural Network
DT	Decision Tree
DWP	Dimension-Wise Probability
EEG	Electroencephalogram
EHR	Electronic Health Record
FID	Fréchet Inception Distance
FPR	False Positive Rate
GAN	Generative Adversarial Network
GBC	Gradient Boosting Classifier
IS	Inception Score

KNN K-Nearest Neighbors

LR Logistic Regression

MA Minibatch Averaging

MD Minibatch Discrimination

MLP Multilayer Perceptron

MMD Maximum Mean Discrepancy

NLP Natural Language Processing

NS Non-Saturating (update rule)

RF Random Forest

ROC Receiver Operating Characteristic

SDS Siamese Distance Score

SNN Siamese Neural Network

TPR True Positive Rate

VAE Variational Autoencoder

WGAN Wasserstein Generative Adversarial Network

Chapter 1

Introduction

1.1 Motivation

We can see a substantial increase in the amount of data that is gathered and stored. We now have massive datasets stored in all imaginable areas, and the health care domain is no exception [6, 7]. The sheer size of these datasets requires new algorithms to extract valuable data and make sense of them. Researchers worldwide are leveraging intelligent systems and machine learning models to accomplish this goal. This is thanks to hardware improvements and advancements in Artificial Intelligence and Machine Learning.

Deep Learning models are now so advanced and practical that we see them everywhere, from smartphones to smart home appliances. However, for Deep Learning models to achieve high-level performance and quality, we need to provide huge amounts of data to them. This massive data requirement is an issue in all areas, and most of the time, a considerable amount of manual human labor is required to gather the data, label it, and process it for public use. In the healthcare domain, the situation is much worse than in other areas as most of the time health records are private, not public.

Companies use de-identification techniques to remove private data from their datasets to mitigate privacy concerns. However, researchers have shown time and time again that almost all of the de-identification methods are prone to re-identification attacks. Given the

importance of the matter, companies rarely share their datasets with the public, and even when they do so for research and educational use, the process is complicated and takes a long time.

Another technique that has recently surfaced is to generate data that closely resembles actual data with a substantial reduction in the leakage of personal and private data. Unfortunately, Synthetic Data Generation has not gained much traction, and research in this area is usually laser-focused on a specific scenario with small amounts of data. Additionally, there is not much solid evidence on the practicality of the synthetic data that these studies have produced. As stated before, due to lack of availability, we cannot mitigate these issues by using massive amounts of data.

We believe that generating models that take advantage of new deep learning techniques can reduce privacy risks. Our goal is to introduce a novel approach to synthetic health care data generation by leveraging new deep learning models.

1.2 Research Questions

The main goal of this research is to create a generative model that can generate useful synthetic EHR datasets. To achieve our goal and measure our success, we came up with the following research questions.

- **RQ1**: How can we measure realism? What are the characteristics of a real dataset that set it apart from a fake one?
- **RQ2**: How can we improve upon existing synthetic data generation models?
- **RQ3**: How can we produce data good enough to replace real datasets in the training phase of practical AI applications?

1.3 Research Hypotheses

Some research focuses on how we can measure the performance of generative models [8, 9]. We hypothesize that deep learning models could perform better than other AI models in measuring the quality of synthetic datasets.

To answer our other two research questions, we hypothesize that Convolutional Neural Networks are better at capturing the input features compared to Multilayer Perceptron (MLPs) since CNNs can capture features that are positioned close to each other in the input.

Additionally, we expect our model to capture shared statistical characteristics of the input dataset, such as its distribution. We think that the statistical similarities will allow replacement of real datasets with our generated synthetic datasets for construction of artificial intelligence and machine learning models.

1.4 Research Challenges

This research addresses multiple challenges, including the three that follow.

1.4.1 Discrete Data

Generative Adversarial Networks (GANs) [10] are known for their high performance as generator models. However, they do not work well with discrete data [11], and our primary electronic health record dataset consists of discrete data.

1.4.2 Difficulties of Training GANs

GANs are hard to train, and one of the most significant problems that researchers run into when training GANs is that they cannot correctly train them, since GANs are prone to mode-collapse [12]. Additionally, GANs can refuse to converge [13].

1.4.3 Evaluation of Synthetic Data

The other main challenge we faced was the evaluation of synthetic data. There is no gold standard to compare with our findings. To demonstrate our generative model's performance, we needed a way to measure its quality. As stated before, the goal is to generate synthetic data and use that data to train other models. We needed some intelligent models that perform well on real datasets, and our goal was to measure their performance relative to when we trained them using our synthetic dataset. We also needed to find a few statistical metrics to compare our findings with the real dataset, as finding a model that works with real data is not always feasible.

1.5 Research Contributions

In this research, we were able to contribute the following.

- **Introduce a new evaluation metric to measure GANs performance:** We introduced a novel approach to GAN evaluation using discriminative models. We then showed its superior performance by comparing it to other known evaluation metrics such as Fréchet Inception Distance (FID) [14] and Inception Score (IS) [15].
- **Introduce a robust deep learning model that can create high-quality data**

using two datasets: We introduced CorGAN, our novel approach to synthetic data generation using convolutional neural networks in GANs. We then chose different techniques capable of measuring each of our datasets' performance, and then compared the output quality to other models. We demonstrated that CorGAN is capable of creating high-quality datasets, and its performance surpasses other generative models.

- **Further improvements and optimizations on CorGAN:** Lastly, we improved our model and generated better samples in a shorter time by further optimizing some parts of our model and utilizing a technique to prevent mode collapse. We call this improved model SCorGAN.

1.6 Outline

The rest of this thesis is organized as follows:

- In Chapter 2, we review related literature on generative models to offer a comprehensive overview of techniques used in this area.
- In Chapter 3, we introduce our domain agnostic evaluation metric to measure GAN performance. We then show that our proposed novel approach can detect common GAN pitfalls like mode-dropping and mode-invention.
- In Chapter 4, we introduce our main contribution, CorGAN, our proposed generative model that can capture correlated features from the input dataset and create high-quality synthetic datasets.
- In Chapter 5, we introduce further optimizations to CorGAN to improve its training time and performance.

- Finally, in Chapter 6, we conclude this research, going over our findings, contributions, and suggestions for future work.

Chapter 2

Review of Literature

In this research, we focus on generating good synthetic health records. Additionally, we introduce a new metric to evaluate generative models. In this chapter, we go through established and new evaluation methods and then review recent generative models.

2.1 Evaluation Metrics

GANs are a form of deep learning generative models that are still relatively new. Given how popular they are, evaluating their performance is also a hot topic as there are many research articles on new evaluation metrics and comparisons between them [16, 17, 18].

One of the most common and well-known metrics is the *Inception Score (IS)* [15]. However, IS is primarily used in the vision domain and is not a good metric for use in other domains, such as music [19]. It also has some issues which make it unable to provide helpful insight and even render it ineffective [20]. The basic idea behind *IS* is that a good image classifier has a classification ability resembling that of humans. Although some researchers have challenged this assumption [21, 22], other researchers have adopted this idea with different generative models, and observed promising results [23, 24, 25].

Another well-known metric is *Fréchet Inception Distance (FID) Score* [14]. FID can overcome some of the shortcomings of *IS*. For example, it confirms if the generative model is not

producing samples of a specific class. Regarding metrics included in this work, we mainly concentrate on FID and IS, and compare them to our approach.

There are other metrics that measure the performance of generative models. *Annealed Importance Sampling (AIS)* [16] provides a metric for decoder-based models. It evaluates log-likelihoods and uses a bidirectional Monte Carlo method for validation. However, some articles show its deficiencies [26].

Skill Rating [17] also provides a measurement for GAN performance. Skill Rating pits the training model against its past and future, or differently tuned versions. Lastly, *Precision-Recall* [27] is a novel metric that tries to distinguish different ways a GAN fails instead of providing a score.

Beyond the utilization of metrics for the evaluation of GANs, a widely used approach is training and testing a classifier on fake and real data, respectively, to assess the fidelity of the synthetic data [28, 29]. Although practical, such an approach limits the evaluation to the use-case under investigation, and neither the classifier nor the training regime can be generalized to different use-cases.

The majority of evaluation metrics are domain-specific, fail to address mode dropping, or have disadvantages in generalization. For example, when comparing our method with FID, although our approach needs labeled data, it does not need a pre-trained classifier as required in FID. Furthermore, as FID operates on a pre-trained image classifier, its generalizability to other domains is problematic. Accordingly, we propose a domain-agnostic metric to evaluate GANs by utilizing *Siamese Neural Networks (SNNs)* and provide a comprehensive analysis.

2.2 Generative Models

Generative Adversarial Networks (GANs) [10] have become one of the most famous generative models ever since their introduction in 2014. There are several research articles on data generation using GANs [30, 31, 32, 33]. Some of them focus on the privacy aspect of data generation, while some others focus on data-driven approaches to overcome the current lack of publicly available data. We employ GANs in our proposed generative model.

Our work closely follows previous work done on *medGAN* [34]. medGAN utilizes GANs to generate EHR data. Their work leverages multilayer perceptron [35] neural networks and can achieve good results. In our work, we try to improve their model by capturing spatial features of the data using Convolutional Neural Networks.

An issue that makes training GANs hard is mode-collapse. It occurs when the generator collapses and the generator keeps generating the same sample for the whole minibatch. One of the proposed methods to prevent mode collapse is called *Minibatch Discrimination (MD)* [15]. In MD, the discriminator investigates the whole batch and penalizes the generator for generating a batch with low entropy. Additionally, the medGAN [34] authors introduced *Minibatch Averaging (MA)* as an alternative to MD.

2.3 Architecture Improvements

There is a lot of research on GANs. As we mentioned in the research challenges section, GANs can be tricky to train, and therefore, researchers are always trying to improve their performance.

Jacobian Clamping [36] is a technique that improves performance by generator conditioning. *Top-K* training [37] achieves better performance by simply removing the gradient of samples

with low scores by the discriminator. Another work suggests [38] image augmentation can improve GANs' performance in the vision domain.

Our research investigates whether we can improve performance by training a better autoencoder, meaning one that sees fewer errors after reconstructing the input by passing it through the encoder and the decoder. We also apply a similar technique to what was used in medGAN to prevent mode-collapse.

Chapter 3

Evaluation of GANs by Discriminative Models

GANs can model complex multi-dimensional data accurately and produce realistic samples. However, due to their implicit estimation of data distributions, their evaluation is a challenging task. Qualitative visual evaluation is the validation method used by most research efforts associated with tackling this issue. Such approaches do not generalize well beyond the computer vision domain. Since many of those evaluation metrics are proposed and bound to the vision domain, they are challenging to apply to other areas. Quantitative measures are necessary to help guide the training and comparison of different GANs models. In this work, we leverage Siamese neural networks to propose a domain-agnostic evaluation metric [3, 39]:

- It is a qualitative evaluation that is consistent with human evaluation.
- It is robust relative to common GAN issues such as mode dropping and mode invention.
- It does not require any pre-trained classifier.

The empirical results of our work demonstrate the superiority of this method compared to the popular Inception Score. They also are competitive with the FID score.

I would like to acknowledge that the work reported in this chapter was carried out in collaboration with Dr. Amirsina Torfi. Our initial work and results were published in ICPR-2020 [3]

and are included in Dr. Torfi's dissertation [39]. We collaborated closely on this research work, that was carried out in the Digital Library Research Laboratory, which is directed by our common advisor, Dr. Fox. The collaboration included model design, development, testing, and evaluation.

3.1 Introduction

Generative Adversarial Networks (GANs) [10] have gained much attention due to their capability to capture data characteristics, producing fake but realistic samples (Figure 3.1), and their superiority compared to other generative models. Many successful research efforts utilized GANs in different applications such as image super-resolution [40], natural language generation [41], healthcare synthetic data generation [42], style transfer [43], etc. [24, 44, 45, 46].

Figure 3.1: Comparison of real (left figure) and fake images (right figure). The fake images are generated using WGAN [1] for training regime and DCGAN [2] as the architecture. These images are taken from [3].

Despite GANs' success for generative purposes, a significant challenge is their quantitative

evaluation because it is almost impossible to understand the underlying data distribution that is captured by a GAN model. Log-Likelihood and Kullback-Leibler divergence are widely used to evaluate generator models focused on density estimation [47]. However, it can be hard to calculate the log-likelihood [47]. Even if we can do so, research shows that it can be misleading for high-dimensional complex data [47].

Many researchers have proposed different GANs evaluation approaches. The majority of research efforts conducted associated with the image domain utilize the visual evaluation of the synthesized sample. Such visual approaches are subjective and might even be misleading [27, 48]. Furthermore, in other areas such as Natural Language Processing, visually evaluating the fake data is not straightforward or even plausible. Indeed, the generalizability of GANs evaluation metrics is a challenging topic of discussion, and only a few research efforts have been conducted regarding domain agnostic evaluation metrics [27, 49].

In the computer vision domain, efforts led to the invention of excellent metrics such as Inception Score (IS) [15] and Fréchet Inception Distance (FID) [14]. However, IS is proven to have many weaknesses such as suboptimality, issues regarding its application beyond the ImageNet dataset [50] (see Figure 3.2), and not being sensitive to mode dropping [20]. The FID score remedies the majority of IS issues. However, it is still not designed to operate beyond the vision domain without modification.

Motivated by such problems, we propose a novel approach to evaluate the GANs using *Siamese Neural Networks (SNNs)*. These are known to be effective discriminative models for verification applications. In the past, researchers have used SNNs to augment the training of GANs [51]. In this work, we leverage SNNs to quantify the evaluation of GANs.

Our Contributions: We **(1)** introduce a novel metric to evaluate the GANs, demonstrating how it works and what desired characteristics it has; **(2)** compare our method with two

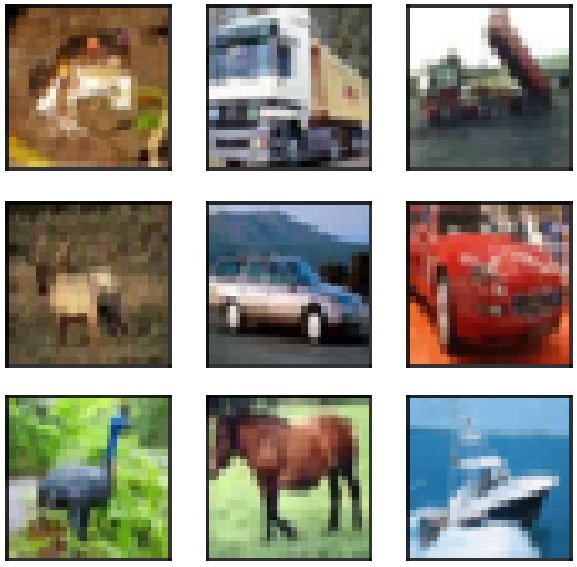


Figure 3.2: The prediction of Inception Score on images from Cifar-10 (black) and the real class labels (red). Clearly IS score fails to go beyond ImageNet. From left to right and top to bottom: Amphibian (Frog), Milk Can (Truck), Milk Can (Truck), Threshing Machine (Deer), Sorrel (Automobile), Sorrel (Automobile), Container Ship (Bird), Japanese Spaniel (Horse), Fox Squirrel (Ship). This image is taken from [3].

other widely used approaches to clarify our model's advantages; and **(3)** demonstrate that our method is domain agnostic by going beyond the vision domain.

3.2 Proposed Approach

How can we train a discriminative model to measure the realistic characteristics of fake data? We propose to train a discriminator using real data. This discriminator is independent of the discriminator trained by the GAN; we call it a Siamese discriminator. For this aim, we employ Siamese Neural Networks.

As a human observes a new pattern, he/she usually can recognize and associate this pattern with an already known concept with a reasonable level of confidence. In evaluating GANs,

this is a challenging task due to the lack of supervised information about the generated data since neither the label nor the explicit prior distributions are available. However, given the discriminative model paradigm, we aim to leverage the real data's implicit distribution to distinguish the generated data's realistic characteristics $p_X(x)$. By realistic, we refer to the similarity of $p_g(x)$ to $p_X(x)$. Thus, the goal here is to quantify this realism.

We train a discriminative model via a supervised learning paradigm with Siamese neural networks, and then we reuse the trained model to measure how well the GAN works. A critical aspect of this quality assessment is the quality measurement of the generated samples. The main focus here is character and image recognition. However, this approach is not bound to the data domain as we go beyond the image domain to further illustrate this desired characteristic of our proposed approach. The utilized model captures the similarity and dissimilarity between inter-class and intra-class samples without considering domain-specific knowledge.

3.2.1 Siamese Architecture

The discriminative model uses a Siamese architecture [52, 53], which consists of two identical neural networks. The aim is to create a target feature subspace for discriminating between similar and dissimilar pairs based on a simple distance metric.

The model is depicted in Figure 3.3. The general idea is that when two samples belong to a genuine pair (a pair in which both samples belong to the same category), their distance in the target feature subspace should be as small as possible, while for an impostor pair (a pair in which the samples belong to different categories), the samples should be as far apart as possible in the output space. Let X_{p_1} and X_{p_2} be a pair of samples as the system's input, whether in training or testing mode. The distance between a pair of samples in the target

subspace is defined as $D_W(X_{p_1}, X_{p_2})$ (e.g., the $\ell_2 - norm$ between two vectors), in which W is the parameters of the whole network (weights).

$$D_W(X_{p_1}, X_{p_1}) = ||F_W(X_{p_1}) - F_W(X_{p_2})||_2. \tag{3.1}$$

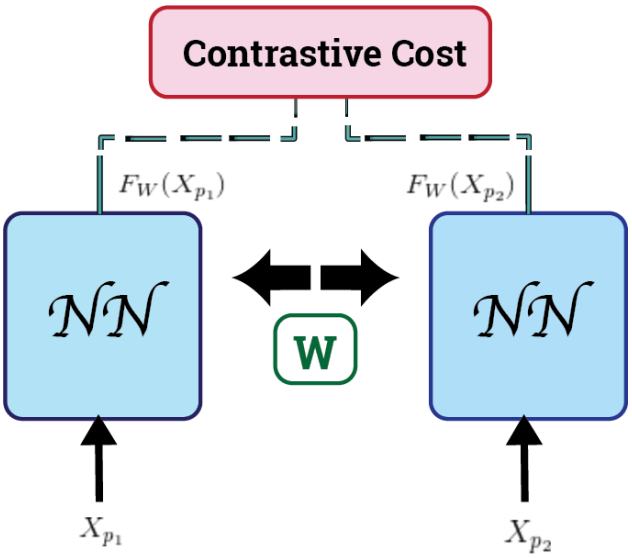


Figure 3.3: Siamese Architecture, Image taken from [3].

3.2.2 Learning

Contrastive Cost

The goal of the loss function $\mathcal{L}_W(X, Y)$ is to minimize the loss in both scenarios, i.e., of encountering genuine and impostor pairs, so the definition should satisfy two conditions, as follows:

$$\mathcal{L}_W(X, Y) = \frac{1}{\mathcal{N}} \sum_{k=1}^{N} L_W(Y_i, (X_{p_1}, X_{p_2})_i), \tag{3.2}$$

where $\mathcal{N}$ is the number of samples for training and the function $\mathcal{L}_W(Y_i, (X_{p_1}, X_{p_2})_i)$ is defined as:

$$\begin{aligned}\mathcal{L}_W(Y_i, (X_{p_1}, X_{p_2})_k) = Y_i * \mathcal{L}_{gen}(D_W(X_{p_1}, X_{p_2})_k) \\ + (1 - Y_i) * \mathcal{L}_{imp}(D_W(X_{p_1}, X_{p_2})_k,\end{aligned} \tag{3.3}$$

where L_{gen} and L_{imp} are defined as:

$$\begin{cases}\mathcal{L}_{gen}(D_W) = \frac{1}{2}(D_W)^2 \\ \mathcal{L}_{imp}(D_W) = \frac{1}{2}(\max\{0, M - D_W\})^2.\end{cases} \tag{3.4}$$

where M is a margin that is obtained by cross-validation, and D_W stands for $D_W(X_{p_1}, X_{p_2})$. Moreover, the *max* argument declares that in the case of an impostor pair, if the distance in the target feature space is greater than the threshold M, there would be no loss.

Input

For training input, we must have genuine and impostor pairs from the real data. The goal of training a Siamese architecture is to put genuine and impostor pairs into close and distant manifolds. To create genuine pairs, we combine samples from the same classes as follows:

$$(X_{p_1}, X_{p_2}) \mid X_{p_1}, X_{p_2} \in \mathbf{y}_i$$

On the other hand, to create imposter pairs, we do as follows:

$$(X_{p_1}, X_{p_2}) \mid X_{p_1} \in \mathbf{y}_i, X_{p_2} \in \mathbf{y}_j, i \neq j$$

3.2.3 Method Statement

Assume having an unconditional sample generation setting with a dataset that is composed of samples $\mathbf{x}^{(i)}$. In unconditional sample generation, the labels $\mathbf{y}^{(i)}$ do not play a role in image generation. However, we need the ground-truth labels to train our Siamese architecture. But, how and why?

As is explained earlier, a Siamese architecture utilizes two identical networks (networks with the same architectures and parameters θ) to create a nonlinear mapping from its input domain to a shared Euclidean output feature space:

$$\psi : \mathcal{X} \rightarrow \mathbb{R}^m$$

Due to the weight sharing and contrastive cost, such a scheme guarantees that:

- similar samples will stay close in the output feature space;
- the model is robust against intra-class samples variations, as the model minimizes intra-class differences;
- dissimilar samples will be placed in distant places in the output space; and
- the model is robust against inter-class sample similarities as the model maximizes inter-class differences.

Hence, we ideally will have separate clusters, in each of which there are samples from just one particular category. For example, all images belonging to dogs will be placed in one cluster, while pictures of cats will reside in another cluster in the output Euclidean space. *Each cluster is a data manifold that belongs to that class.*

Such a learning paradigm will create a system that can:

1. recognize the category of a fake sample, i.e., which cluster of data the fake sample belongs to (determining the closest cluster to classify the fake sample);
2. determine how close the fake sample is to the cluster or any real data within (determining the quality of a fake sample given the real samples); and
3. clarify how diverse the generated fake samples are (to penalize the model for mode collapse).

This metric would punish the model if it does not deliver all data distribution modes (comparing with the data inside clusters) and classes (all clusters have some fake samples associated with them). How close a fake sample is to a manifold determines the *precision*. On the other hand, *recall* refers to how well the generator can produce samples similar to the variety of samples in the data manifold [54].

3.2.4 Siamese Distance Score (SDS)

Once the network is trained, we can use it for evaluation purposes. Referring back to Figure 3.3, we technically have one set of weights **W** and two copies of one network. Evaluation of *a fake sample* has the following procedure:

1. We feed all real samples to the trained neural network and compute the feature vector

$F_W(X_i^r)$ for each real sample X_i^r. If we have N real samples, we will have N feature vectors:

$$\mathcal{F}_i^r = F_W(X_i^r), i \in \{1, \ldots, N\}$$

2. We feed a fake (synthesized) sample X_j^s to the network and compute its output feature vector $F_W(X_j^s) = \mathcal{F}_j^s$. If we have M fake samples, we will have M feature vectors:

$$\mathcal{F}_j^s = F_W(X_j^s), j \in \{1, \ldots, M\}$$

3. We calculate the Euclidean distance of each $\mathcal{F}_j^s$ with all previously calculated feature vectors $\mathcal{F}_i^r$ ($i \in \{1, \ldots, N\}$), and we pick the K closest ones (K smallest distances $\mathcal{D}_i^j$ and j is fixed for each fake sample). The index list is denoted with $\mathcal{P}$ for which $|\mathcal{P}| = K, \mathcal{P} \in \{1, \ldots, N\}$.

$$\mathcal{D}_i^j = \left\| \mathcal{F}_j^s - \mathcal{F}_i^r \right\|, i \in \{1, \ldots, N\}$$

4. Among the closest K $\mathcal{D}_i^j$ distances, we pick their associated $\mathcal{F}_i^r$. From that, we extract their associated real samples and labels.

$$\mathcal{D}_{\mathcal{P}}^j \Rightarrow \mathcal{F}_r^{\mathcal{P}} \Rightarrow X_r^{\mathcal{P}}, |\mathcal{P}| = K, \mathcal{P} \in \{1, \ldots, N\}$$

5. As we have the class of real samples, using a simple majority vote (K-nearest neighbor algorithm), we determine the class of the fake sample. Basically, the majority vote operates on $X_r^{\mathcal{P}}$ samples (real samples with indexes of $\mathcal{P}$) and their associated classes.

6. After we determine the class of the fake sample as C, in the cluster of K nearest samples, we take out the real samples that have the determined class label of the

fake sample (class C). The index list of these samples is denoted with $\mathcal{R}$ for which $|\mathcal{R}| = R, \mathcal{R} \in \{1, \ldots, N\}$.

$$X_r^{\mathcal{R}}, \mathbf{y}(X_r^{\mathcal{R}}) = C$$

7. We calculate the distance of the fake sample output feature F_s^j with all $F_r^{\mathcal{R}}$ feature vectors, then compute the average, and denote it as SDS_j. The subscript j refers to the fake sample index. We then average SDS_j over all fake samples and call it the *Siamese Distance Score (SDS)*.

3.3 Experiments

To conduct our experiments, we split our data into three partitions as $\mathcal{D} = \mathcal{G} \cup \mathcal{S} \cup \mathcal{E}$. $\mathcal{G}$, $\mathcal{S}$, $\mathcal{E}$ will be used for training our generative model, discriminative model, and evaluations, respectively. It is worth noting that **(1)** $\mathcal{G} \cap \mathcal{S} \cap \mathcal{E} = \varnothing$, **(2)** both $\mathcal{G}$ and $\mathcal{S}$ follow the same data distribution due to the random partitioning, and **(3)** all class labels in the data are available in $\mathcal{G}$ and $\mathcal{S}$, in balance.

We denote the fake generated data as $\mathcal{F}$. We run all of our experiments ten times and then report the average. Such a setup ensures the robustness of the results against possible sensitivity regarding the network weights. We observed high stability and very low variance. This indicates the robustness of the model relative to small variations of the network weights due to retraining which corrects a major drawback of the Inception Score [20].

3.3.1 Image Domain

We made the following decisions for our image domain studies to ensure a balanced collection of experiments: We used three common datasets in the GANs literature: MNIST [55],

Fashion-MNIST [56], and CIFAR-10 [57]. To train our generative model, we used WGAN [1], due to its stability and robustness to mode collapse [15], with DCGAN [2] as the architecture for all datasets. We fixed the latent dimension (input noise z) to 100 and the noise distribution to be $\mathcal{N}(0, 1)$. The batch size used is 64. We train our GAN for 20, 20, and 100[1] epochs for MNIST, Fashion-MNIST, and CIFAR-10, respectively.

We used a three-layer convolutional neural network to train our discriminative Siamese network, followed by a fully connected layer of size 1024 as the output embedding space. To boost the training process, we used batch normalization [58]. We utilized LeakyReLU [59] for layers' activation to avoid encountering dead ReLUs and zero gradients. In this work, to calculate SDS, we used a simple nearest neighbor (K-nearest neighbor in which $K = 1$) classifier as we did not observe a unique difference by picking $K > 1$.

Quality

To assess the visual quality, we demonstrate the evaluation of fake samples on MNIST and Fashion-MNIST in Figure 3.4. As can be observed, as the SDS increases, the visual image quality decreases. Such observation aligns with human evaluation.

Mode dropping and invention

To evaluate *mode dropping* (not generating a class of data) and *mode invention* (generating samples that do not belong to any class), we used MNIST, Fashion-MNIST, and CIFAR-10 datasets by fixing all their images' sizes to 32×32. Each dataset has ten classes. Here's how we set up our experiments:

- We pick $\mathcal{S}$ so it only has 5 classes and train our Siamese model with it.

[1]Required number epochs for each model to generate samples that looked good to a human observer

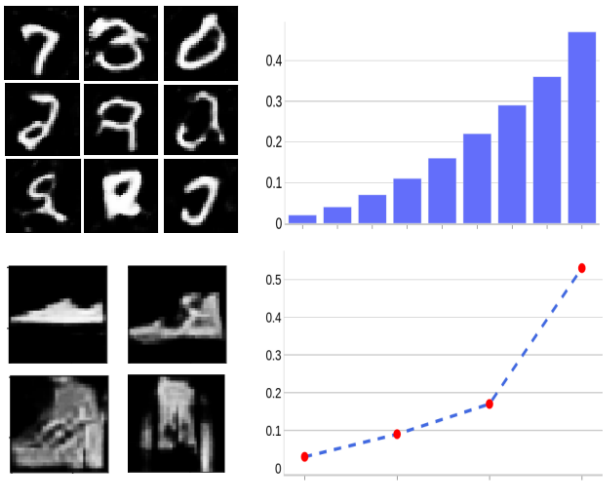


Figure 3.4: Demonstration of the quality measurement of fake samples which is consistent with human evaluation. The horizontal axis is the index of images from left to right and top to bottom. The vertical axis is the SDS score of the sample. Image is taken from [3].

- We create a test set $\mathcal{T}$ from $\mathcal{E}$ which only has i classes.
- For each i we measure SDS.

The results are depicted in Figure 3.5. As can be observed, increasing the number of classes from 1 to 5 gives a better score, as fewer classes mean mode dropping. Additionally, increasing the number of classes beyond five results in an increase in SDS, which shows the detection of mode invention.

Figure 3.6 demonstrates the comparison of our method SDS with the other two popular methods in the literature. SDS and FID are both good at capturing mode dropping. However, SDS shows higher sensitivity in mode dropping.

Intra-class mode-collapse

As we mentioned earlier, the intra-class mode collapse (mode dropping) refers to the situation that GAN can generate all modes (classes) of data but only one or few examples from each

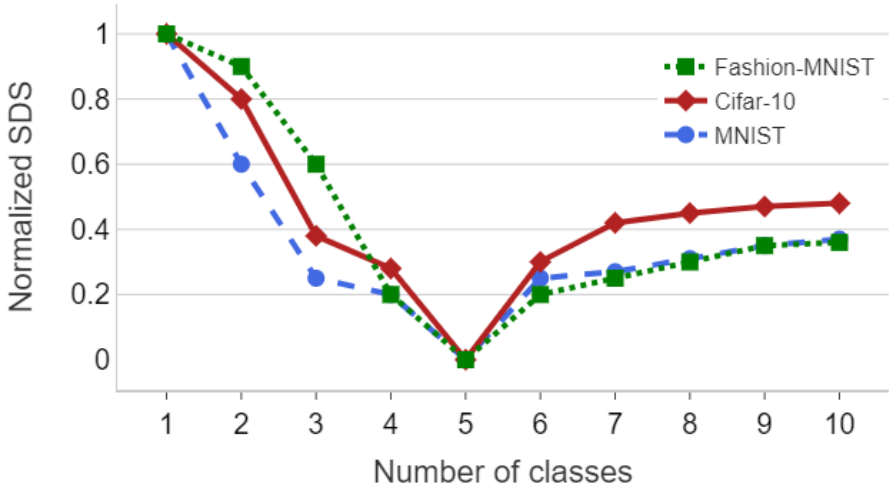


Figure 3.5: SDS is sensitive to mode dropping and mode invention. The higher the SDS is, the worst the results are. The score for each dataset has its scale. For each dataset experiment, the scores are normalized according to their own max-min obtained scores. Image taken from [3].

mode (fails to generate a variety within class samples). To assess SDS metric robustness to this phenomena, we conduct the following experiments:

- We pick $\mathcal{S}$ so it has all classes and train our Siamese model with it.
- We create a test set $\mathcal{T}$ from $\mathcal{E}$ which also has all classes. However, only p percentage of each class's samples are used (note that $\mathcal{S}$ and $\mathcal{T}$ are mutually exclusive and $|\mathcal{T}| = |\mathcal{E}| \times p$).
- For each class, regardless of p, we select an identical number of samples (say $\mathcal{K}$ number of samples per class). Note that $\mathcal{K}$ is maximum when $p = 1$. We select $\mathcal{K}$ as $\mathcal{K} = |\mathcal{T}|/\mathcal{C}$ in which $\mathcal{C}$ is the number of classes.
- For each p we measure SDS and report the results.

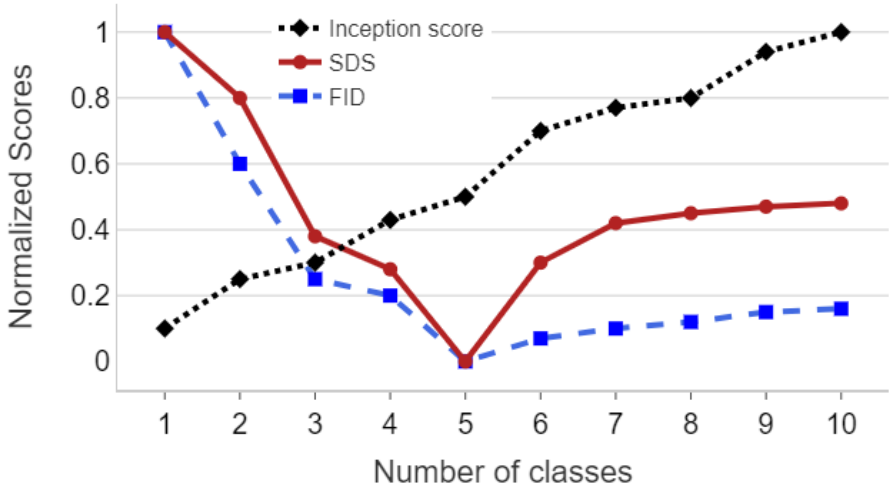


Figure 3.6: Comparison of *Inception Score*, *FID*, and our proposed metric *SDS*. The score for each dataset has its own scale. For each dataset experiment, its scores are normalized according to its own max-min obtained scores. Image taken from [3].

The results are depicted in Figure 3.7. The metric is sensitive to mode dropping. Based on the evaluation approach, the nearest neighbor algorithm is forced to pick a sample as the closest one. If there are only a few modes available, the algorithm picks from the samples that do not represent the variety of the samples available in $\mathcal{T}$. Henceforth, such behavior was expected, which is one of the desired characteristics of the proposed approach.

3.3.2 GANs Models Evaluation

To investigate the feasibility of using our metric for different GANs models' comparison, we focus on the work's model evaluation aspects. For the baseline, we used the GAN with the non-saturating update rule (NS) as proposed by Goodfellow [10].

We analyze three different generally utilized GAN models on the Cifar10 dataset: **(1)** Base-

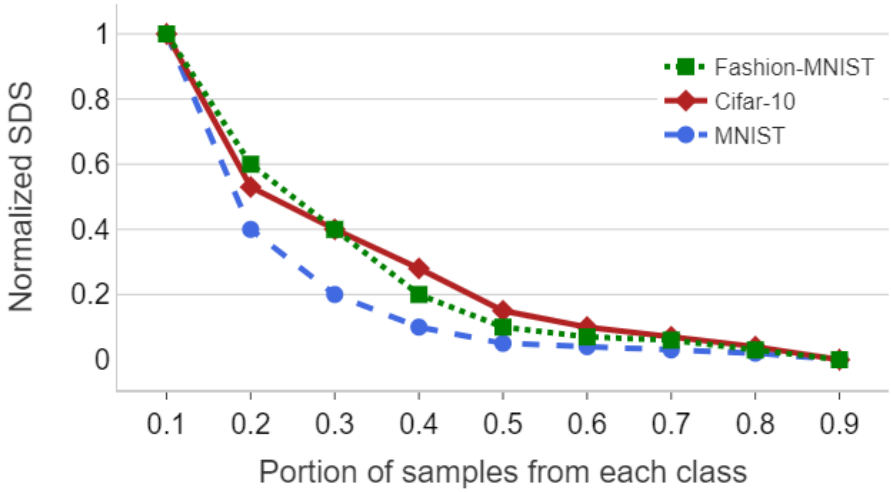


Figure 3.7: SDS is sensitive to intra-class mode dropping. The higher the SDS is, the worse the results are. The score for each dataset has its scale. For each dataset experiment, the scores are normalized according to its own max-min obtained scores. Image taken from [3].

line GAN [10], **(2)** GAN with spectral normalization (SN) [60], and **(3)** WGAN with gradient penalty (GP) [61]. The utilized architecture is DCGAN. The results are given in Table 3.1.

Table 3.1: The comparison of different GANs models using the FID and SDS metrics. These results were previously published in [3].

	FID	SDS
Baseline	53.72 ± 5.43	0.53 ± 0.08
SN	31.25 ± 2.17	0.15 ± 0.03
WGAN + GP	38.48 ± 2.73	0.21 ± 0.04

In concurrence with the results obtained in [60] to rank the models, SDS also reports the same GANs ranking based on their generated sample quality and consistent with FID. We computed both FID and SDS scores from 10,000 real and generated samples.

3.3.3 Utilization of the Approach Beyond the Image Domain

To assess the generalizability of our proposed metric, we considered the healthcare domain and the Electronic Health Records (EHRs) data, which have very different statistics and characteristics compared to image data.

We performed our experiments with the UCI Epileptic Seizure Recognition dataset [62]. This dataset contains brain activities; the main objective is brain seizure classification. Approximately 20% of the samples are classified as seizure activity (we have only two class labels). The number of features for each sample is 179 and there is a total of 11500 samples. The first 178 features are associated with the Electroencephalogram (EEG) values, while the last one is the class label.

Although the main goal of this work is to evaluate GANs, our proposed approach can be extended to evaluate any kind of generative model. To showcase that, we picked three different successful generative models in the healthcare domain – *Variational Autoencoders* [63], *medGAN* [34], and *CorGAN* [5] – to generate the synthetic one-dimensional data:

- **Variational Autoencoder (VAE):** A 1D convolutional neural network is utilized with two hidden layers of 128 for both the encoder and the decoder.
- **medGAN:** The medGAN [34] architecture includes: **(1)** fully-connected layers, **(2)** shortcut connections to augment the generator, and **(3)** minibatch-averaging [34] to overcome mode-collapse. Although medGAN has been originally proposed for generating discrete records, we removed its autoencoder part to directly generate continuous records.
- **CorGAN:** The CorGAN [5] architecture has the following elements: **(1)** 1-D convolutional neural network for discriminator and generator, **(2)** WGAN training regime.

Both medGAN and CorGAN have proven to be successful methods in generating discrete and continuous healthcare records.

We evaluate the averaged SDS score after training our Siamese architecture with three fully connected layers, each with the size of 128. We report the results of our score along with Maximum Mean Discrepancy (MMD) [64], which is known to be an effective sample-based GANs evaluation metric [65]. For both MMD and SDS, an equal number of synthetic and real samples are chosen for calculation. The results are depicted in Figure 3.8. As can be observed, SDS is consistent with MMD regarding the comparison of the three models.

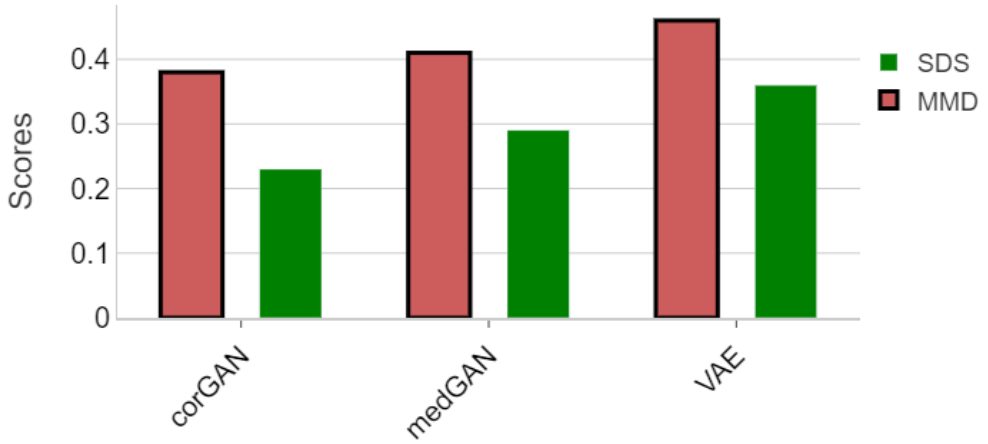


Figure 3.8: The comparison between different generative models using MMD and SDS. As can be seen, our proposed metric can effectively rank generative models concur with the MMD score. Image taken from [3].

3.4 Conclusion

We proposed an evaluation metric for GANs, which relies on Siamese neural networks. Our metric can be applied in the evaluation of any generative model. The proposed approach enables us to evaluate GANs without the need for any pre-trained classifiers. We empirically proved that our method could address different GANs failure situations (mode dropping, mode invention, intra-mode collapse) and is sensitive to visual quality aligned with human evaluation. Finally, the proposed approach's significant advantage is its domain agnostic characteristics that make it useful beyond the image domain and in various applications.

Chapter 4

CorGAN: Correlation-Capturing Convolutional GANs

In this chapter, we describe our novel approach to generating synthetic health records. We leverage Convolutional GANs, which are mainstream in the computer vision domain, and are able to create remarkable samples. We would like to note that this chapter was done in collaboration with Dr. Amirsina Torfi. Earlier versions appeared in Virginia Tech's institutional repository [5], The Thirty-Third International Flairs Conference [4], and Dr. Torfi's dissertation [39]. We both developed the architecture and I carried out experiments we used to evaluate this model. We made the following contributions:

- We propose our novel generative model. In our model, CorGAN [5], we use Convolutional Autoencoders (CAs) and Convolutional GANs. We show that CorGAN is capable of generating both discrete and continuous data.
- We demonstrate that Convolutional Neural Networks (CNNs) are proficient in detecting the correlation between features.
- We demonstrate that the synthetic data CorGAN generates is a good substitute for real data in classification.
- We employ the membership inference attack on CorGAN to evaluate the privacy of our proposed model.

4.1 Method

In this section, we review the structure of an EHR dataset and the architecture of our proposed model. It is important to know about the EHR dataset, as the generative model is trained using that dataset and it generates synthetic data in the same space as the input space.

4.1.1 EHR Data Structure

We primarily use the MIMIC-III dataset [66, 67]. In this dataset, each row represents a patient and each column shows whether a patient has been diagnosed with a medical condition or not (ICD-9 [68] code). We chose this dataset because it is the most common dataset used in EHR data generation models that is publicly available.

4.1.2 Architecture

Our proposed model [4] uses GANs as its base architecture. As we mentioned in Chapter 2, our model follows the work done in medGAN [34]. In this section we describe medGAN's [34] architecture and we follow it by describing the architecture of our proposed generative model.

Baselines

The combination of a generator and a discriminator creates a GAN. At each iteration, the generator creates a new sample and sends it to the discriminator. The discriminator then attempts to decide if the item is a fake or a genuine sample from the input dataset.

During the training phase, the generator tries to create better samples to fool the discrim-

inator by learning the real data distribution. In comparison, the discriminator tries to distinguish these better but still fake samples from real ones. We continue the training process until the discriminator can no longer distinguish generated samples from the real ones.

Even though GANs are very capable, they are not so in all areas. An issue with GANs is that they are not capable of generating good discrete datasets. This proves to be a big issue as we have a lot of discrete data in the healthcare domain. Discrete data makes the backpropagation process inaccurate, which results in impossible or very hard to train GANs. To overcome this, we developed our model by using the general architecture of medGAN [34].

Figure 4.1 shows CorGAN's architecture. Our baseline model utilized Multilayered Perceptrons (MLPs) for both the generator and the discriminator, which mirrors the model used in medGAN [34].

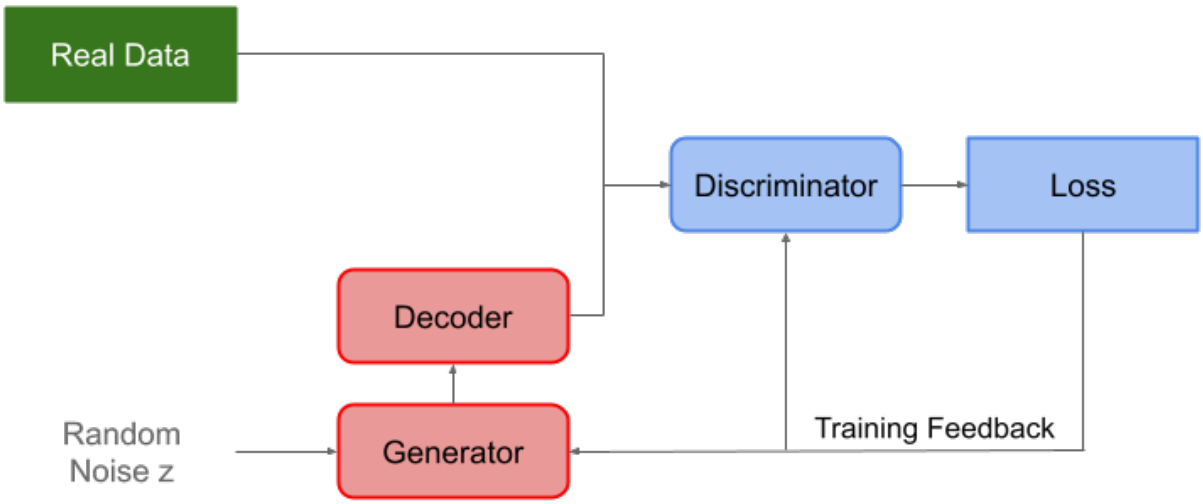


Figure 4.1: The proposed architecture for CorGAN.

$\boldsymbol{X}$ describes real input, which is discrete EHR data. $\boldsymbol{z}$ describes the random distribution that the generator ($\boldsymbol{G}$) takes as its source. $\boldsymbol{Dec}$ is the decoder part of our autoencoder model, which maps the continuous data in the latent space to the discrete data of our input space. The discriminator $\boldsymbol{D}$ takes the real data $\boldsymbol{X}$ and fake data $\boldsymbol{Dec(G(z))}$ and tries to

distinguish fake input from real input.

As we mentioned before, GANs do not work well with discrete data. Researchers have proposed a few ways to overcome the limited capability of GANs in generating discrete data [69, 70, 71, 72]. We choose an indirect approach that trains the model in the continuous space and then transfers it into the input space by employing autoencoders. Autoencoders consist of two neural networks, as shown in Figure 4.2: an encoder and a decoder.

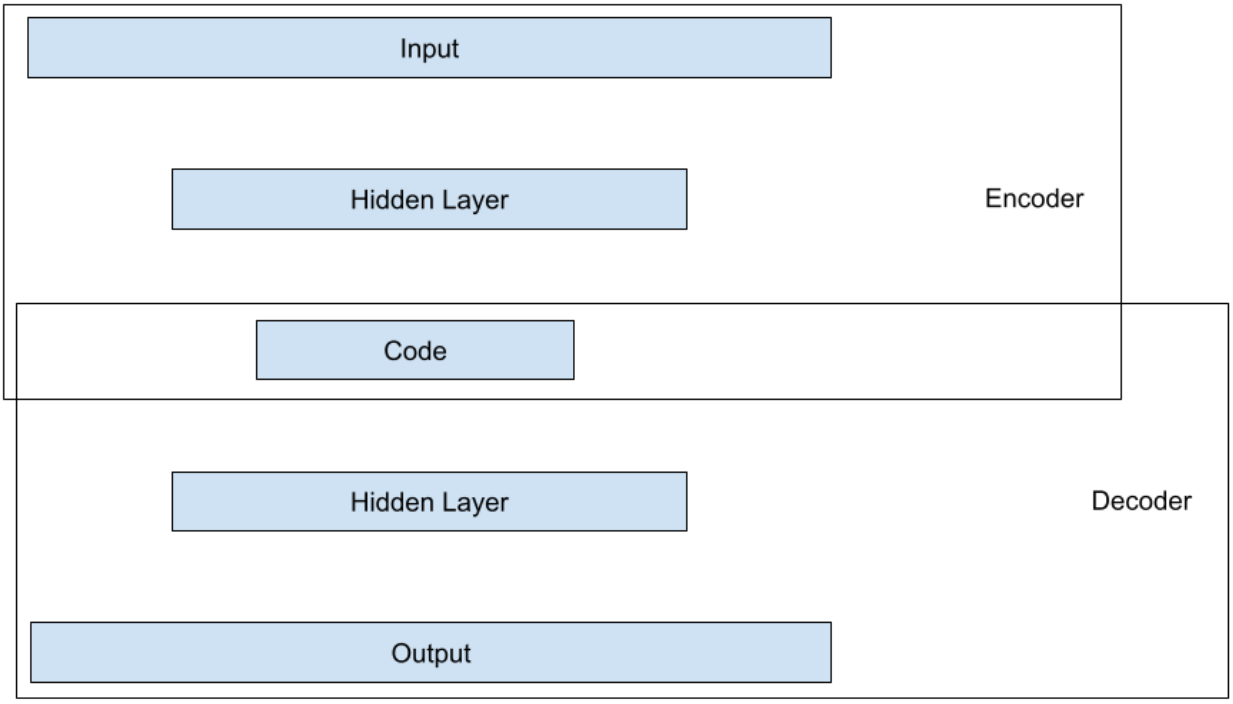


Figure 4.2: The architecture of an Autoencoder.

The autoencoder's encoder takes discrete input and then maps it to a corresponding continuous space. The decoder part of the autoencoder takes that continuous data and maps it back to the input space (Equation 4.1). We use the Binary Cross Entropy (BCE) loss function in the training process to calculate the network's loss (Equation 4.2) [5].

$$y_i = \boldsymbol{Dec}(\boldsymbol{Enc}(x_i)) \tag{4.1}$$

$$BCE_{loss} = -\frac{1}{N}\sum_{i=1}^{N} x_i \log(y_i) + (1 - x_i)\log(1 - y_i) \tag{4.2}$$

After training, the autoencoder should output data that closely resembles the source input data. Our model does not need the encoder, so we take the decoder and put it after the generator. In this model, the generator takes the random distribution, generates a batch of data, and then passes it to the decoder. The decoder will convert that data into the EHR input space. The discriminator will then train on that batch.

CorGAN Architecture

Our generative model is called CorGAN. In CorGAN, we replaced our baseline MLP networks with one-dimensional convolution neural networks (1D CNNs). We then used a one-dimensional convolutional autoencoder (1D CAE) to help capture spatial features of the dataset and create a better mapping between our one-dimensional and discrete input data and our continuous latent space.

4.2 Discriminative Model

This section describes our approach to train a discriminative model that can distinguish a real dataset from a fake one. Since we have trained the discriminator of our GAN with its generator, it is already being fooled into believing our fake data is real, so we cannot use that discriminator. We propose to use a discriminator that is trained using real and synthetic

data independently from our generative model.

4.2.1 Siamese Architecture

We leverage a Siamese model [73] as our discriminative model. Siamese models use two identical neural networks in parallel with each other. We will feed those two models a pair and calculate the distance between the output of those models. We need to build a target feature space to distinguish similar and contrasting pairs. If we feed our model two samples from the same pool, then the distance between them in the target feature space should be very close, while if we use two samples from different pools, their distance in the target feature space should indicate they are quite far from each other. Our model is pictured in Figure 4.3.

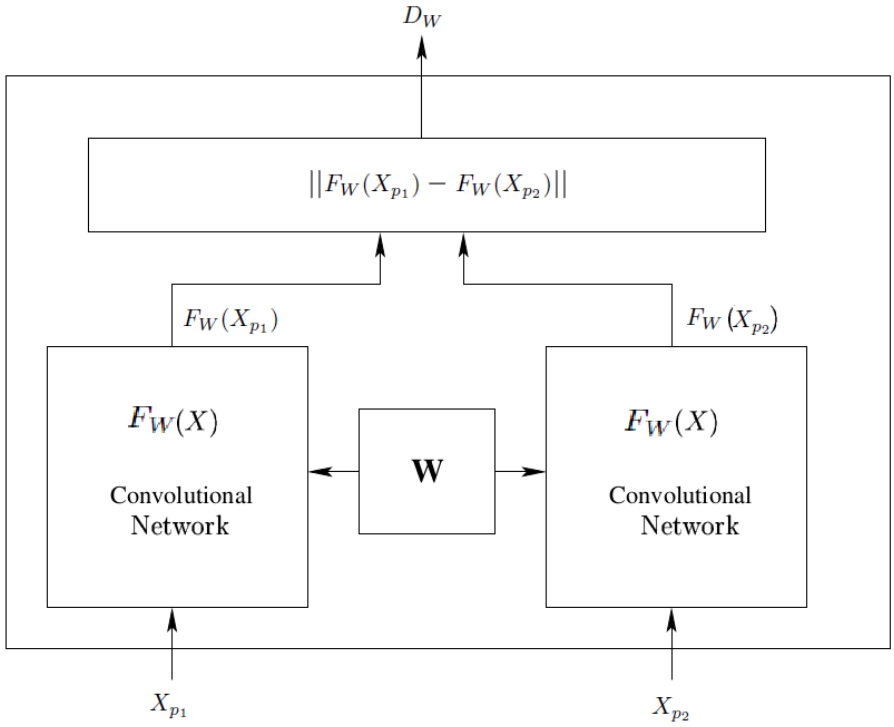


Figure 4.3: Siamese Model. Image taken from [4].

We feed our model two samples we call X_{p_1} and X_{p_2}. We call the distance between them in the feature space $D_W(X_{p_1}, X_{p_2})$, also known as the contrastive loss function. W represents the weights of the discriminator models. When we feed our model a genuine pair, $D_W(X_{p_1}, X_{p_2})$ should be low, and for impostor pairs, it should be high. The label (Y) should be one if both our samples are genuine and should be zero otherwise. F denotes our discriminator model, and W (weights vector) is the same across discriminators as we use two identical discriminators. We can calculate the distance using the following equation:

$$D_W(X_{p_1}, X_{p_1}) = ||F_W(X_{p_1}) - F_W(X_{p_2})||_2 \tag{4.3}$$

4.2.2 Contrastive Cost

The following is the equation for the loss function. It is applied to both genuine and impostor pairs, so it should attempt to minimize the cost in either case.

$$L_W(X, Y) = \frac{1}{N} \sum_{i=1}^{N} L_W(Y_i, (X_{p_1}, X_{p_2})_i) \tag{4.4}$$

In the previous equation, N denotes the total number of samples, and i denotes the index of the current sample. We define $L_W(Y_i, (X_{p_1}, X_{p_2})_i)$ as follows [74]:

$$\begin{aligned} L_W(Y_i, (X_{p_1}, X_{p_2})_i) &= Y * L_{gen}(D_W(X_{p_1}, X_{p_2})_i) \\ &+ (1 - Y) * L_{imp}(D_W(X_{p_1}, X_{p_2})_i) + \lambda ||W||_2 \end{aligned} \tag{4.5}$$

The regularization parameter (λ) is used in the last term. L_{gen} and L_{imp} are defined as the functions of $D_W(X_{p_1}, X_{p_2})$ as follows [74]:

$$\begin{cases} L_{gen}(D_W(X_{p_1}, X_{p_2})) = \frac{1}{2} D_W(X_{p_1}, X_{p_2})^2 \\ L_{imp}(D_W(X_{p_1}, X_{p_2})) = \frac{1}{2} max\{0, (M - D_W(X_{p_1}, X_{p_2}))\}^2 \end{cases} \tag{4.6}$$

We employ cross-validation to calculate the margin parameter M. It is clear that in this equation, if the distance in the target space of an impostor pair becomes larger than the margin parameter M, we reduce it to zero.

4.3 Experiments

This section covers our experiments. We start by describing our experiments' setup. We then go over methods we used to improve and better train our generative model.

4.3.1 Setup

To generate discrete EHR data, we used MIMIC-III [66, 67], a known and publicly available dataset. MIMIC-III has health records of nearly 50 thousand patients. We used a table in it that shows patients and ICD-9 codes [68] that they have been diagnosed with. There are 1071 unique ICD-9 codes, and they each show whether or not the patient was diagnosed with a specific condition. In our preprocessing, we make a single binary table of 46,000×1071 where rows represent patients, and the columns represent ICD-9 codes.

We implemented five different models, including CorGAN. All of the implemented models have the same architecture. MA denotes Minibatch Averaging [34], MD denotes Minibatch Discrimination [15], and BN denotes Batch Normalization [58]. Table 4.1 shows the details of the implemented models. MLP networks we used have two fully connected layers. The first one has a size of 256, while the second one has a size of 128. We used the same MLP

network in all models that leverage this kind of neural network to keep our results consistent and comparable.

Table 4.1: Comparison of different baseline architectures.

Name	Decoder (pre-trained)	Generator	Technique
GAN	Autoencoder (NO)	MLP	Regular Training
GAN_{pre}	Autoencoder (YES)	MLP	Regular Training
GAN_{pre}	Autoencoder (YES)	MLP	MD
$medGAN$	Autoencoder (YES)	MLP	MA + BN
CorGan	Autoencoder (YES)	1-D CNN	MD + BN

We developed CorGAN using PyTorch 1.3 [75]. In the training process, our batch size was 1,000 while we used the Adam Optimizer [76] with a learning rate of 1e-3. Our autoencoder is fully convolutional, meaning that the encoder uses convolutional operations to extract features, and the decoder uses deconvolutional [77] operations to reconstruct data. Our autoencoder uses a 256-dimensional latent space.

4.3.2 Mode-Collapse

As we mentioned previously, mode collapse is one of the biggest pitfalls of GANs. It happens when the generator limits itself to a specific set of parameters and keeps generating similar batches. Since the discriminator has no way of providing feedback to the generator not to generate the same sample, the mode collapse happens, and the generator keeps generating the same samples.

We utilized minibatch discrimination (MD) [15] to prevent mode-collapse in our experiments. MD penalizes the generator for generating a batch with low variety.

4.3.3 Training Improvements

To further improve our generative model, we employed Wasserstein GAN (WGAN) [1] as our training algorithm as the WGAN [1] authors showed improvement with their generative models in more recent research articles [78, 79, 80]. Wasserstein-1 distance is defined as the cost to transform the generated dataset's distribution $\boldsymbol{P_g}$ to the real dataset's distribution $\boldsymbol{P_r}$ (Equation 4.7 [1]).

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \Pi(\mathbb{P}_r, \mathbb{P}_g)} E_{(x,y)\sim\gamma}\big[\, \|x - y\| \,\big] \tag{4.7}$$

$\Pi(\mathbb{P}_r, \mathbb{P}_g)$ shows all of the possible distributions in the previous equation, and $\gamma(x, y)$ refers to the cost of transforming $\boldsymbol{P_r}$ to $\boldsymbol{P_g}$. However, it is hard to calculate the infimum in this equation. The researchers behind WGAN proposed the following (Equation 4.8) based on Kantorovich-Rubinstein duality [1, 81].

$$W(\mathbb{P}_r, \mathbb{P}_\theta) = \sup_{\|f\|_L \leq 1} E_{x\sim\mathbb{P}_r}[f(x)] - E_{x\sim\mathbb{P}_\theta}[f(x)] \tag{4.8}$$

In this equation, the supremum is calculated over all the 1-Lipschitz functions [81] $f : \mathcal{X} \to \mathbb{R}$. To help make things easier, we call the greatest lower bound the infimum and the least upper bound supremum.

K-Lipschitz functions show the following behavior:

$$\forall(x_1, x_2) \in \mathbb{X}, \exists K \in \mathbb{R} : d_Y(f(x_1), f(x_2)) \leq K d_X(x_1, x_2) \tag{4.9}$$

Using $K = 1$ and Equation 4.9 we get the following equation:

$$\forall(x_1, x_2) \in \mathbb{X} : |f(x_1) - f(x_2)| \leq |x_1 - x_2| \tag{4.10}$$

We can see that we need a 1-Lipschitz function to calculate the cost. We can deploy a neural network to learn it. To do so, we need to create our discriminator model without using a sigmoid function. Additionally, we need to return a scalar rather than the probability. The WGAN authors [1] suggest a simple gradient clipping method to limit the weights matrix in the model to enforce the restriction.

4.3.4 Evaluation Methods

To evaluate our model and the synthetic data it generates, we primarily leverage the two following metrics: dimension-wise probability (DWP) and maximum mean discrepancy (MMD). During this process, we divide our real dataset into two sets: $\mathbb{S}_{tr}$ and $\mathbb{S}_{te}$. In the training process, we use $\mathbb{S}_{tr}$. After properly training our model, we use it to generate a synthetic dataset $\mathbb{S}_{syn}$ with the same size as $\mathbb{S}_{te}$.

- **Dimension-wise probability:** This is a simple metric to verify that the probability of seeing an ICD-9 code is the same in both our synthetic dataset and the real dataset. We refer to the Bernoulli success probability of each dimension when each ICD-9 code is a dimension.
- **Maximum Mean Discrepancy (MMD) [82]:** This measures similarity between the distribution of two datasets. To evaluate our synthetic data, we use a slightly different version called Kernel Maximum Mean Discrepancy [82], and we measure its value for our synthetic data and our real testing dataset.
- **Discriminative Model:** This is the Siamese model we described in the previous sec-

tion. We separately train this discriminator model and then run our real and synthetic dataset through it and present our findings.

4.3.5 Dimension-wise Probability

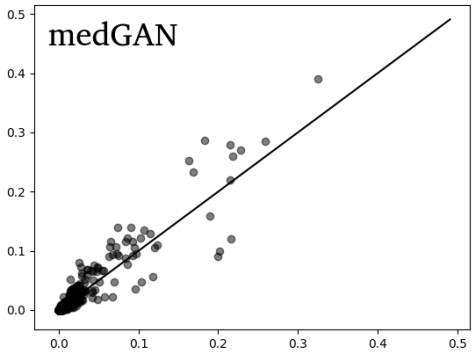
We show only our top three performing models in Figure 4.4. Each point in the scatter plots represents an ICD-9 code. The closer the points are to the $\boldsymbol{y} = \boldsymbol{x}$ line, the better. CorGAN and WGAN performed similarly, while they both outperformed medGAN.

4.3.6 Maximum Mean Discrepancy

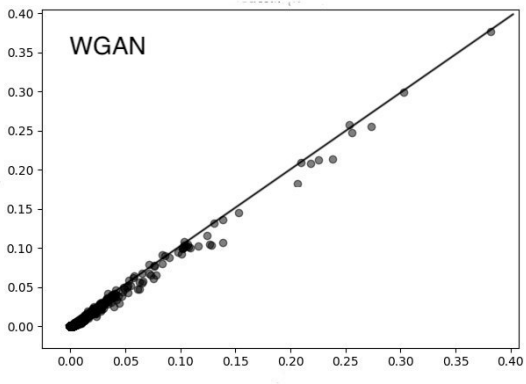
We report the results of our experiments in Table 4.2. We generate a dataset of synthetic data for each model and then compare it to the real dataset. If we compare the real dataset to itself, the results should be zero. We are reporting our findings with the mean and standard deviation of five experiments for each model.

Table 4.2: Results of employing MMD on different models with the real dataset. Results taken from [5].

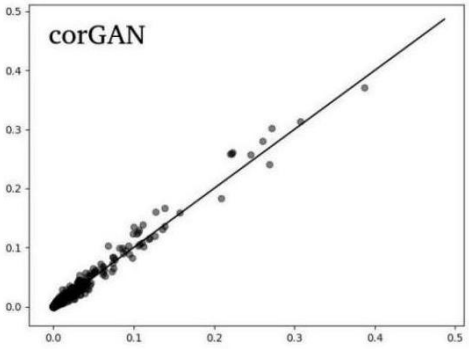
Name	Score
GAN	0.0064 ± 0.00035
GAN_{pre}	0.0048 ± 0.00022
GAN_{pre+md}	0.0043 ± 0.00018
$medGAN$	0.0032 ± 0.00021
$WGAN$	0.0018 ± 0.00024
$\boldsymbol{CorGan}$	$\mathbf{0.0008 \pm 0.00015}$



(a) medGAN



(b) WGAN



(c) CorGAN

Figure 4.4: Results of dimension-wise probability analysis. Each point represents one ICD-9 code. The x-axis represents the real dataset, and the y-axis denotes the synthetic dataset. The diagonal line displays the perfect scenario where they both have the same probability. Images taken from [5].

4.3.7 Discriminator Model

As we mentioned before, to properly train and evaluate our model, we split our training dataset into two parts: $\mathbb{S}_{tr}$ and $\mathbb{S}_{te}$. We used our model to generate a synthetic dataset $\mathbb{S}_{syn}$ for evaluation after the training process. We need to create pairs (X_{p1}, X_{p2}) to train and evaluate our discriminative model. In the training phase, to create a genuine pair, we pick both X_{p1} and X_{p2} from the $\mathbb{S}_{tr}$. To create an imposter pair, we pick one sample from $\mathbb{S}_{tr}$

and the other from $\mathbb{S}_{te}$. In the evaluation phase, we pick one sample from $\mathbb{S}_{syn}$ and the other from $\mathbb{S}_{tr}$ for a genuine pair. For the imposter pair, we pick one sample from $\mathbb{S}_{syn}$ and the other from $\mathbb{S}_{te}$.

We used Receiver Operating Characteristic (ROC) [83] to evaluate the results from this experiment. Figure 4.5 shows our results. The vertical axis represents True Positive Rate (TPR) while the horizontal axis represents False Positive Rate (FPR). We can see that our discriminative model can match our synthetic data with its real counterparts, and displays outstanding performance.

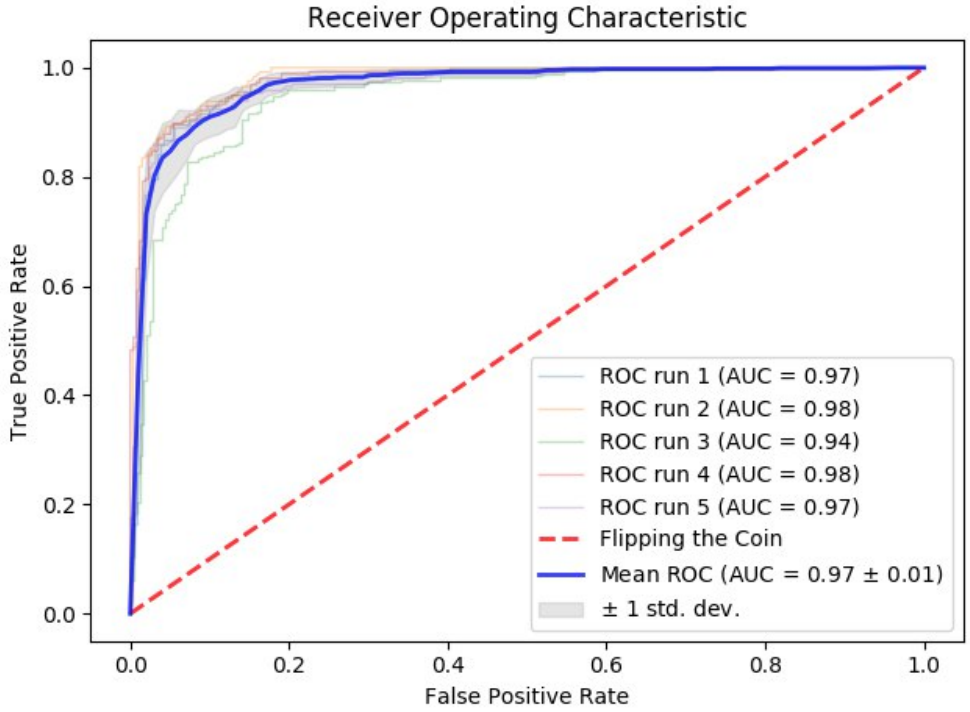


Figure 4.5: Results of running the discriminative model five times. Image taken from [5].

4.4 Privacy Assessment

We used the membership inference attack [84] to determine if a dataset is leaking information about its participants. This attack is usually done to find the similarity of two datasets. We are concerned about leaking private information of patients whose records were used to train our generative model. One possible risk is that an attacker might be able to use re-identification attacks [85] to identify specific patients used in training. We conducted this experiment to investigate the similarity of our synthetic data with the training dataset. In a perfect scenario, the attacker is not able to match any of the synthetic records with any record from the training dataset.

In our evaluation, we have two pools of patient records. One consists of synthetic patient records P_{syn} and the other consists of real patient records $\mathbb{U}$. Half of $\mathbb{U}$ is taken from the training dataset $\mathbb{S}_{tr}$ and the other half from the testing dataset $\mathbb{S}_{te}$, which we denote P_{tr} and P_{te}, respectively. P_{syn} has 10,240 samples and they all come from the $\mathbb{S}_{syn}$ dataset. We used the same records in P_{syn} and $\mathbb{U}$ for all of our experiments.

If our model was leaking data, it would leak data from S_{tr}, as we never used $\mathbb{S}_{te}$ samples in the training process. To investigate this, we calculated the cosine similarity between synthetic samples and the samples we provided to the attacker. We then flag results that are higher than a preset threshold as a match, which we denote as $\mathbb{M}$. To calculate the value of this threshold, we used a normal distribution with a mean of 0.5 and a standard deviation of 0.01 and selected 100 random samples from it. We report the results from the best attack.

We define precision as the number of synthetic records that the attacker inferred were used in training, i.e., those matched with a sample from $\mathbb{P}_{tr}$, divided by the total number of matched records (Equation 4.11). We define recall as the fraction of the training dataset's members that were correctly identified by the attacker as used in training, i.e., the number

of synthetic records that we matched with a sample from $\mathbb{P}_{tr}$, divided by the total number of records in the training dataset (Equation 4.12). We report the precision and recall metrics from our experiment. Table 4.3 shows our findings. It shows that as we increase the number of real records used in the attack, the attack becomes less accurate. This shows that the similarity between synthetic and real datasets is limited to a subset of the training dataset. We can use this attack and remove matched synthetic records from our synthetic dataset to get to a smaller synthetic dataset, that according to this metric, is not similar to our training dataset, minimizing risk of re-identification attacks using similar methods. This experiment can be done with other generative models to compare them with CorGAN.

$$precision = \frac{|\mathbb{M} \cap \mathbb{P}_{tr}|}{|\mathbb{M}|} \tag{4.11}$$

$$recall = \frac{|\mathbb{P}_{tr} \cap \mathbb{M}|}{|\mathbb{P}_{tr}|} \tag{4.12}$$

Table 4.3: Results of employing Membership Inference Attack. Published in [5].

$\mathbb{U}$	100	1k	2k	3k	4k	5k
Precision	0.60	0.51	0.41	0.40	0.40	0.39
Recall	0.05	0.10	0.19	0.28	0.27	0.28

4.5 Conclusion

We proposed CorGAN, our novel approach to generate synthetic health records. To even further understand the association between input features, CorGAN employs one-dimensional convolutional neural networks. CorGAN generates better samples than our baseline versions, as discussed in the previous sections. This is a noticeable improvement that shows the

advantage of using CNNs over MLP networks to capture correlated data. In addition, we introduce a metric for measuring the similarity of real and synthetic samples. Our experiments demonstrate that the evaluation metric effectively matches synthetic samples with the corresponding real data that we used to train the generative model. Lastly, we do a membership inference attack on our synthetic data and we show that our generative model does not leak information from the training samples. We can filter out problems and have a synthetic dataset that does not leak private patient records.

Chapter 5

SCorGAN: Improved Correlation-Capturing Convolutional GANs

This chapter describes further improvements we made on top of our previous generative model (CorGAN). Our experiments showed that the autoencoder plays a huge role in CorGAN's performance. In SCorGAN, we improve the autoencoder significantly and introduce a modified version of minibatch averaging to improve our training time and sample quality. We made the following contributions:

- We propose SCorGAN, our improved generative model based on CorGAN, that generates better samples while being two times faster than CorGAN in the training phase.
- Improved Autoencoder: We improve the autoencoder significantly, meaning that this new autoencoder has a better mapping between our input space and the latent space. It has four times less error when we compare it to our previous autoencoder.
- Minibatch Subsampling: We introduce a modified version of the minibatch averaging introduced in medGAN [34]. In this method, we leverage a technique that can capture spatial features of the input and helps the discriminator distinguish synthetic samples from the real ones.

- We demonstrate that using an MLP neural network in conjunction with CNNs can help capture the correlations between features not spatially adjacent in the input data.
- We demonstrate that SCorGAN can generate high-fidelity samples using our base datasets (MIMIC-III [66, 67] and UCI Epileptic Seizure Recognition Dataset [62]). We then demonstrate that we can use these synthetic samples as the training dataset to train competent classifiers that work with real datasets.

5.1 Method

This section first reviews the new dataset we used, the UCI Epileptic Seizure Recognition Dataset [62]. We then introduce our mode-collapse prevention method, Minibatch Subsampling. Lastly, we review the architectural improvements we made to our generative model.

5.1.1 New EHR Dataset

In addition to MIMIC-III [66, 67], we used another dataset in our experiments, the UCI Epileptic Seizure Recognition dataset [62], which is the same dataset we considered in Section 3.3.3. In this dataset, each row is the patient's brain activity (electroencephalogram or EEG) except for the last column, representing the label associated with the patient, which indicates whether they were having a seizure or not. Further, the UCI Epileptic Seizure Recognition Dataset [62] is a time-series dataset. On the other hand, MIMIC-III [66, 67] is an adjacency matrix connecting patients to ICD-9 codes. We believe the difference between these two datasets makes them good candidates to evaluate our generative model. Additionally, both of these datasets are well known and widely discussed in the literature, which should help people compare our approach to other approaches in this area.

5.1.2 Minibatch Subsampling

Researchers behind medGAN [34] proposed a new approach to prevent mode-collapse called Minibatch Averaging (MA). In this technique, before feeding a minibatch to the discriminator, we calculate the average of each feature in the minibatch and then add this average to the input as additional features for all samples in the minibatch, practically doubling the number of features. We propose a slightly modified version of this technique, and we downsample the minibatch to one-tenth of its features and then add that to the minibatch, increasing the number of features by ten percent. To downsample a minibatch, for every 10 features, we calculate the average of those 10 features in the minibatch and then add that value to every sample in the minibatch as an additional feature, as shown in Figure 5.1. We hypothesized that one of the primary benefits of minibatch averaging was that fake batches start with an average close to 0.5, that can be very different from the average of real samples. Minibatch Subsampling should help the discriminator identify fake samples and, therefore, boost the training process. We employed this technique in our proposed model to great effect (see the experiments sections 5.2.3, 5.2.4 and 5.2.5).

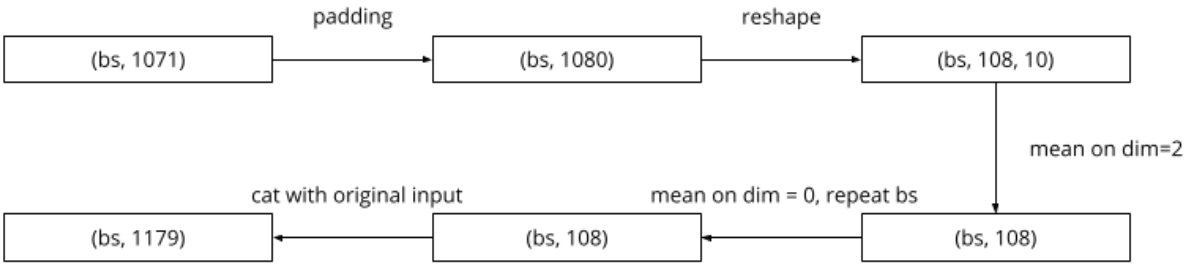


Figure 5.1: Minibatch Subsampling on MIMIC-III; bs denotes batch size.

5.1.3 Early Architectures

We noticed that the Convolutional Neural Networks (CNNs) were getting a lot of attention in the computer vision domain while none of the famous medical data generation models were using CNNs [34, 86, 87]. In CorGAN, we used deep convolutional neural networks in the autoencoder, discriminator, and generator. We surveyed other generative models in the health care domain, e.g., our baseline research medGAN [34] along with others like PATE-GAN [88], DPGAN [87], medWGAN [86], and medBGAN [86], all of which leveraged MLP neural networks to a great extent. In CorGAN's development process, we started replacing MLP networks with CNNs, one network at a time, to see the effect of using CNNs. We started with the autoencoder, and in our testing with the MIMIC-III [66, 67] dataset, we noticed that our new autoencoder is on par or better than medGAN's autoencoder since it had a slightly better error rate compared to medGAN. Creating the generator and discriminator using CNN networks proved problematic as we failed to train our early models. We focused on the generator and found out that we could not train it because it was not deep enough. By increasing the number of layers to 6, we trained our model and had better results (using dimension-wise probability evaluation) than our baseline medGAN model. Following a similar procedure for the discriminator, we obtained our final CorGAN model.

5.1.4 Architectural Improvements

We hypothesized that perhaps using MLP layers after convolutional neural networks can simplify our models and speed up the training process, to make further improvements. We also hypothesized that the correlation between medical records could be easily captured using a two-layer CNN. Our baseline model [34] used one-layer MLP networks for both encoder and decoder. To develop our improved autoencoder, we started developing an autoencoder using

deep MLP neural networks to evaluate their performance, since we had no data on how deep MLP autoencoders would perform. In our experiments, however, deep MLP autoencoders had a higher error rate than our baseline model [34], resulting in us abandoning further experimentation on deep MLP autoencoders. We created the new encoder and decoder using a two-layer CNN network connected to a two-layer MLP network to test our hypothesis. To compare the autoencoders, we used the MIMIC-III [66, 67] validation dataset. We fed each model with the validation dataset, and we counted the number of digits that changed after going through both the encoder and the decoder. A perfect autoencoder would have zero changed digits. Our experiments showed that both medGAN and CorGAN autoencoders have about 30K errors, while our best-performing experimental model had less than 9K errors in the same setting. In SCorGAN, we employ this improved autoencoder.

Our generator is the same as CorGAN, while our discriminator is slightly modified to count for the extra features that minibatch subsampling adds to the input data. We employed two Conv1D with kernel sizes of 11 and 7 for the first one and the second one, respectively, and then two fully connected layers to the latent space for the encoder in our autoencoder. We used two fully connected layers and then two ConvTranspose1D (with the same settings as Conv1D layers in the encoder) layers in the decoder.

5.1.5 Adapting to UCI Dataset

To adapt our model to generate UCI Epileptic data, we tried to use the exact same CNN networks in all parts of the generative model. We used the same two CNN networks in the autoencoder for both the encoder and the decoder. However, the UCI Epileptic dataset has only 178 features compared to the MIMIC-III dataset's 1071. We wanted to have the same latent space of size 128 for both models. This lead us to reduce the size of our MLP

networks in the encoder and the decoder. The size of the hidden layer between those two MLP networks stayed the same between both models at 256. Since we used the same size for the latent space for both datasets, we used the same generator for both models. However, our input features are of different sizes and we had to make changes to the discriminator. To adapt our discriminator model, we had to reduce the size of our model. The output of the discriminator is a bit denoting if the record is fake or genuine. Since the UCI Epileptic Seizure Recognition dataset has far fewer features, we needed to reduce our model size to get to the same one bit output. To achieve this and adapt our model to the UCI dataset, we used the same first three layers of our discriminator. We removed the fourth layer and reduced the last layer's kernel size from 3 to 2.

5.2 Experiments

In this section, we start by describing how we ran our experiments. Then we briefly describe the evaluation metrics we used. Lastly, we go over the results of our experiments and conclusions.

5.2.1 Setup

In the SCorGAN experiments, we used the MIMIC-III dataset [66, 67] as well as the UCI Epileptic Seizure Recognition dataset [62]. The process for the MIMIC-III dataset was the same as in the CorGAN experiments.

We described the UCI Epileptic Seizure Recognition dataset [62] in the previous section. It has fewer features than MIMIC-III [66, 67], so we modified our model slightly to accommodate the new data structure. It is worth mentioning that the UCI dataset consists of

floating-point numbers, so instead of binary cross-entropy, we used mean square error as the loss function to train the autoencoder.

We developed SCorGAN in PyTorch 1.7. We used the batch of 64 with the Adam Optimizer (same as CorGAN).

For evaluation, we compared SCorGAN to CorGAN and medGAN, to assess our new generative model, relative to the baseline models.

5.2.2 Evaluation Methods

To evaluate the performance of our model with the MIMIC-III and UCI Epileptic Seizure datasets, we employ the following evaluation metrics. Just like our previous work, we divide both of our datasets into $\mathbb{S}_{tr}$ and $\mathbb{S}_{te}$ for training and testing, respectively. After training it, we use it to generate a synthetic dataset $\mathbb{S}_{syn}$ with the same size as $\mathbb{S}_{tr}$.

- **Dimension-wise probability:** This is the same metric we used to evaluate CorGAN. We use it again to test SCorGAN with MIMIC-III. The probability of seeing an ICD-9 code in a patient should be close to seeing it in the real dataset.
- **Maximum Mean Discrepancy (MMD) [82]:** This is the same metric we used to evaluate CorGAN. This metric measures similarity between the distribution of two datasets. To evaluate our synthetic data, we use a slightly different version called Kernel Maximum Mean Discrepancy [82], and we measure its value for our synthetic data and our real testing dataset. Please note that these values were gathered using different experiments and are not comparable with MMD results of the previous chapter.
- **Binary Classifier:** We use this metric to evaluate SCorGAN with the UCI Epileptic

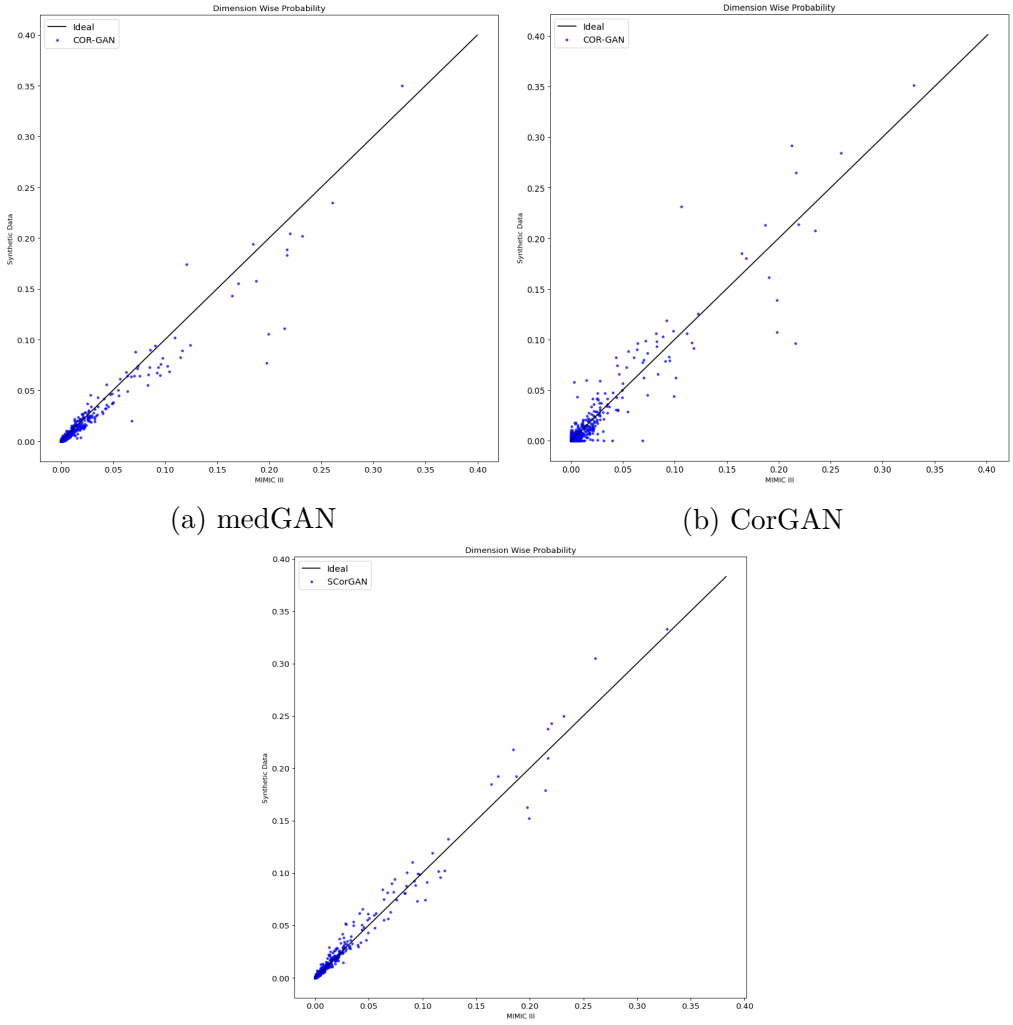
Seizure Dataset. We use multiple known classifier models, and we train them using $\mathbb{S}_{syn}$. Then we evaluate the precision and recall of those classifiers when we use them on $\mathbb{S}_{te}$.

5.2.3 Dimension-wise Probability

Figure 5.2 shows dimension-wise probability results of medGAN, CorGAN, and SCorGAN. Each point in the scatter plots represents an ICD-9 code with MIMIC-III. The closer the points are to the $\boldsymbol{y} = \boldsymbol{x}$ line, the better. All models are trained for 100 epochs using the same settings. SCorGAN noticeably outperforms CorGAN and medGAN. In our testing, dimension-wise probability results of CorGAN improve a great deal if it is trained for 300 epochs; see Figure 5.3. However, SCorGAN does not benefit from more training epochs but it (see Figure 5.2.c and Figure 5.3.a) still outperforms a well-trained CorGAN. One of the important parts of this graph is near the origin point; SCorGAN dots are closer to the $\boldsymbol{y} = \boldsymbol{x}$ line and are further away from the $\boldsymbol{X}$ axis and the $\boldsymbol{Y}$ axis. This shows that SCorGAN can capture the features that rarely occur better than CorGAN and medGAN.

5.2.4 Maximum Mean Discrepancy

We report the results of our experiments with the MIMIC-III and UCI Epileptic Seizure Recognition datasets in Tables 5.1 and 5.2, respectively. We generate a dataset of synthetic data for each model and then compare it to the real dataset. If we compare the real dataset to itself, the results should be zero. We are reporting our findings with the mean and standard deviation of five experiments for each model. We only compared medGAN, CorGAN, and SCorGAN as we have already established that both medGAN and CorGAN outperform our baseline models.



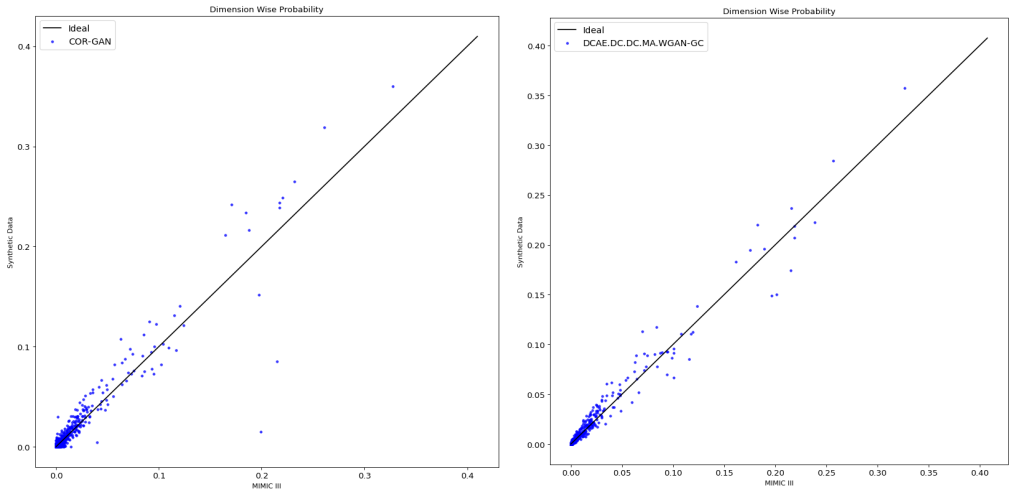
(a) medGAN (b) CorGAN

(c) SCorGAN

Figure 5.2: Results of dimension-wise probability analysis with MIMIC-III. Each point represents one ICD-9 code. The x-axis represents the real dataset, and the y-axis denotes the synthetic dataset. The diagonal line displays the perfect scenario where they both have the same probability. Models are trained for 100 epochs.

5.2.5 Binary Classifier

To evaluate the practicality of the generated synthetic dataset, we created this experiment using binary classifiers. Since this is a binary classification task, we were limited to the UCI Epileptic Seizure Recognition Dataset [62] as the MIMIC-III [66, 67] does not have a binary



(a) CorGAN 300 Epochs (b) SCorGAN 300 Epochs

Figure 5.3: Results of dimension-wise probability analysis with MIMIC-III. Models are trained for 300 epochs.

Table 5.1: Results of employing MMD on synthetic data of medGAN, CorGAN and SCorGAN with the MIMIC-III dataset.

Name	MMD Score
medGAN	0.00105 ± 0.00014
CorGan	0.00125 ± 0.00018
SCorGan	$\mathbf{0.00046 \pm 0.00012}$

label associated with its records. We developed K-NearestNeighbors (KNN) [89], Random Forest (RF) [90], Logistic Regression (LR), Decision Tree (DT), XGBoost Classifier [91], and Gradient Boosting Classifier (GBC) models for the UCI dataset. Table 5.3 shows the results of SCorGAN compared to CorGAN and medGAN. SCorGAN outperforms both the CorGAN and medGAN generative models in all of the metrics using all classifiers. Please note that having an abysmal recall value makes a poor generative model as we would like to identify all patients experiencing a seizure. The results are not as good as with the real dataset, but they are very close, and they present a remarkably good substitute for the real data.

Table 5.2: Results of employing MMD on synthetic data of medGAN, CorGAN and SCorGAN with the UCI Epileptic Seizure Recognition dataset.

Name	MMD Score
medGAN	0.01667 ± 0.00196
CorGan	0.00953 ± 0.00032
SCorGan	$\mathbf{0.00196 \pm 0.00074}$

Table 5.3: Results of employing various binary classifiers on synthetic UCI Epileptic Seizure Recognition data.

	KNN	RF	LR	DT	XGBoost	GBC
AUC (Real)	0.971	0.993	0.502	0.862	0.996	0.983
Precision	0.989	0.896	0.286	0.756	0.898	0.858
Recall	0.263	0.936	0.427	0.838	0.961	0.941
AUC (SCorGAN)	0.921	0.966	0.541	0.710	0.970	0.939
Precision	0.966	0.719	0.268	0.353	0.703	0.698
Recall	0.391	0.927	0.508	0.796	0.933	0.863
AUC (CorGAN)	0.834	0.569	0.494	0.529	0.629	0.595
Precision	0.475	0.249	0.391	0.218	0.265	0.274
Recall	0.754	0.751	0.392	0.574	0.591	0.555
AUC (medGAN)	0.632	0.980	0.506	0.560	0.641	0.668
Precision	1.000	1.000	0.253	0.710	0.750	0.546
Recall	0.003	0.164	0.477	0.468	0.312	0.647

5.3 Conclusion

In this chapter, we presented our improved model that we call SCorGAN. We demonstrated that this new model is better at generating synthetic data than our baseline models, medGAN and CorGAN, using the provided evaluation metrics.

5.3.1 Contributions

We made the following contributions in SCorGAN:

- developed an improved autoencoder;
- introduced a minibatch subsampling technique; and
- developed new models to adapt to the UCI dataset.

We made improvements in the autoencoder model resulting in four times less error. We proposed our method of minibatch subsampling to boost the training time by up to two times. We modified both CorGAN and SCorGAN to generate a different type of EHR dataset, and then we showed that this newer model can achieve significantly better results in the same number of training epochs than our previous model, CorGAN, for both of the MIMIC-III and UCI Epileptic Seizure datasets. Finally, we showed that our synthetic dataset can be used instead of the real dataset to train classifiers that work with real data.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

In this work, we aimed to solve the issue of EHR dataset availability by creating high-quality synthetic data as a replacement for real EHR datasets. We started by determining the research questions and research challenges, and described state-of-the-art generative models and evaluation metrics that are widely used in the literature.

We developed our novel generative model, CorGAN. We demonstrate that it can create high-fidelity datasets in various areas of medical records. We then developed a novel discriminator model to evaluate the fidelity of our synthetic dataset. Additionally, we developed our improved generative model, SCorGAN, by fine-tuning its architecture and employing a new technique to make the training process faster. Lastly, we developed multiple traditional classifiers using our synthetic dataset and showed that it is a practical replacement for real EHR datasets.

6.2 Publications

Our work, and collaborative work in which we have played a significant role, have resulted in four publications [3, 4, 5, 39].

6.3 Future Work

In Chapter 3, we proposed a domain agnostic evaluation metric to assess the quality of synthetic datasets. In addition to being domain-agnostic, our evaluation metric can assess the performance of any generator mode. However, it cannot assess all types of datasets. Our metric employs Siamese neural networks, and it requires labeled datasets in its training phase. Therefore, it cannot be used to evaluate non-labeled synthetic datasets. Developing another approach that can evaluate non-labeled datasets using Siamese networks would be interesting future work. Additionally, we evaluate our model in limited areas. An interesting effort would be applying it in other areas and comparing it to other metrics.

In Chapter 4, we proposed a novel generative model using Convolutional Neural Networks. We then improved it in Chapter 5. We used WGAN in both models. It would be interesting to see how other GANs perform in a similar experiment. Additionally, we employed Convolutional Autoencoders in our generative model. An interesting work would be to use a different model to map input data into a continuous space for GAN training. Lastly, researchers have proposed various methods to improve GAN's training. Future work including these improvement techniques on top of our model could lead to promising results.

Bibliography

[1] Ishaan Gulrajani, Faruk Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville. Improved Training of Wasserstein GANs. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5767–5777, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/892c3b1c6dccd52936e27cbd0ff683d6-Abstract.html.

[2] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. URL http://arxiv.org/abs/1511.06434.

[3] Amirsina Torfi, Mohammadreza Beyki, and Edward A. Fox. On the Evaluation of Generative Adversarial Networks By Discriminative Models. In *25th International Conference on Pattern Recognition, ICPR 2020, Virtual Event / Milan, Italy, January 10-15, 2021*, pages 991–998. IEEE, 2020. doi: 10.1109/ICPR48806.2021.9412214. URL https://doi.org/10.1109/ICPR48806.2021.9412214.

[4] Amirsina Torfi and Edward A. Fox. CorGAN: Correlation-Capturing Convolutional Generative Adversarial Networks for Generating Synthetic Healthcare Records. In Roman Barták and Eric Bell, editors, *Proceedings of the Thirty-Third International Florida Artificial Intelligence Research Society Conference, Originally to be held in North Mi-*

ami Beach, Florida, USA, May 17-20, 2020, pages 335–340. AAAI Press, 2020. URL https://aaai.org/ocs/index.php/FLAIRS/FLAIRS20/paper/view/18458.

[5] Amirsina Torfi and Mohammadreza Beyki. Generating Synthetic Healthcare Records Using Convolutional Generative Adversarial Networks. *Team term project report for CS6604: Digital Libraries, Virginia Tech, Blacksburg, VA*, 2019. URL http://hdl.handle.net/10919/96186.

[6] David Gans, John Kralewski, Terry Hammons, and Bryan Dowd. Medical Groups' Adoption Of Electronic Health Records And Information Systems. *Health Affairs*, 24 (5):1323–1333, 2005. doi: 10.1377/hlthaff.24.5.1323. URL https://doi.org/10.1377/hlthaff.24.5.1323. PMID: 16162580.

[7] Eric W. Ford, Nir Menachemi, and M. Thad Phillips. Research Paper: Predicting the Adoption of Electronic Health Records by Physicians: When Will Health Care be Paperless? *J. Am. Medical Informatics Assoc.*, 13:106–112, 2006. doi: 10.1197/jamia.M1913. URL https://doi.org/10.1197/jamia.M1913.

[8] James Jordon, Jinsung Yoon, and Mihaela van der Schaar. Measuring the quality of synthetic data for use in competitions. *CoRR*, abs/1806.11345, 2018. URL http://arxiv.org/abs/1806.11345.

[9] Linda Moniz, Anna L. Buczak, Lang Hung, Steven Babin, Michael Dorko, and Joseph Lombardo. Construction and Validation of Synthetic Electronic Medical Records. *Online Journal of Public Health Informatics*, Dec. 2009. doi: 10.5210/ojphi.v1i1.2720. URL https://doi.org/10.5210/ojphi.v1i1.2720.

[10] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative Adversarial Nets. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D.

Lawrence, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13, 2014, Montreal, Quebec, Canada*, pages 2672–2680, 2014. URL https://proceedings.neurips.cc/paper/2014/hash/5ca3e9b122f61f8f06494c97b1afccf3-Abstract.html.

[11] R. Devon Hjelm, Athul Paul Jacob, Adam Trischler, Gerry Che, Kyunghyun Cho, and Yoshua Bengio. Boundary Seeking GANs. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=rkTS8lZAb.

[12] Akash Srivastava, Lazar Valkov, Chris Russell, Michael U. Gutmann, and Charles Sutton. VEEGAN: Reducing Mode Collapse in GANs using Implicit Variational Learning. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 3308–3318, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/44a2e0804995faf8d2e3b084a1e2db1d-Abstract.html.

[13] Naveen Kodali, Jacob D. Abernethy, James Hays, and Zsolt Kira. How to Train Your DRAGAN. *CoRR*, abs/1705.07215, 2017. URL http://arxiv.org/abs/1705.07215.

[14] 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 Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Ad-*

vances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA, pages 6626–6637, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/8a1d694707eb0fefe65871369074926d-Abstract.html.

[15] Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved Techniques for Training GANs. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 2226–2234, 2016. URL https://proceedings.neurips.cc/paper/2016/hash/8a3363abe792db2d8761d6403605aeb7-Abstract.html.

[16] Yuhuai Wu, Yuri Burda, Ruslan Salakhutdinov, and Roger B. Grosse. On the Quantitative Analysis of Decoder-Based Generative Models. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=B1M8JF9xx.

[17] Catherine Olsson, Surya Bhupatiraju, Tom B. Brown, Augustus Odena, and Ian J. Goodfellow. Skill Rating for Generative Models. *CoRR*, abs/1808.04888, 2018. URL http://arxiv.org/abs/1808.04888.

[18] Samaneh Azadi, Catherine Olsson, Trevor Darrell, Ian J. Goodfellow, and Augustus Odena. Discriminator Rejection Sampling. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=S1GkToR5tm.

[19] Li-Chia Yang and Alexander Lerch. On the evaluation of generative models in music.

Neural Computing and Applications, 32(9):4773–4784, May 2020. ISSN 1433-3058. doi: 10.1007/s00521-018-3849-7. URL https://doi.org/10.1007/s00521-018-3849-7.

[20] Shane T. Barratt and Rishi Sharma. A Note on the Inception Score. *CoRR*, abs/1801.01973, 2018. URL http://arxiv.org/abs/1801.01973.

[21] Swaminathan Gurumurthy, Ravi Kiran Sarvadevabhatla, and R. Venkatesh Babu. DeLiGAN: Generative Adversarial Networks for Diverse and Limited Data. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 4941–4949. IEEE Computer Society, 2017. doi: 10.1109/CVPR.2017.525. URL https://doi.org/10.1109/CVPR.2017.525.

[22] Zhiming Zhou, Han Cai, Shu Rong, Yuxuan Song, Kan Ren, Weinan Zhang, Jun Wang, and Yong Yu. Activation Maximization Generative Adversarial Nets. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=HyyP33gAZ.

[23] Xun Huang, Yixuan Li, Omid Poursaeed, John E. Hopcroft, and Serge J. Belongie. Stacked Generative Adversarial Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 1866–1875. IEEE Computer Society, 2017. doi: 10.1109/CVPR.2017.202. URL https://doi.org/10.1109/CVPR.2017.202.

[24] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive Growing of GANs for Improved Quality, Stability, and Variation. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=Hk99zCeAb.

[25] Youssef Mroueh and Tom Sercu. Fisher GAN. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 2513–2523, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/07042ac7d03d3b9911a00da43ce0079a-Abstract.html.

[26] Aditya Grover, Manik Dhar, and Stefano Ermon. Flow-GAN: Combining Maximum Likelihood and Adversarial Learning in Generative Models. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 3069–3076. AAAI Press, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17409.

[27] Mehdi S. M. Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. Assessing Generative Models via Precision and Recall. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 5234–5243, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/f7696a9b362ac5a51c3dc8f098b73923-Abstract.html.

[28] Daniel Jiwoong Im, He Ma, Graham W. Taylor, and Kristin Branson. Quantitatively Evaluating GANs With Divergences Proposed for Training. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada,*

April 30 - May 3, 2018, Conference Track Proceedings. OpenReview.net, 2018. URL https://openreview.net/forum?id=SJQHjzZ0-.

[29] David Lopez-Paz and Maxime Oquab. Revisiting Classifier Two-Sample Tests. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=SJkXfE5xx.

[30] Jason A. Walonoski, Mark Kramer, Joseph Nichols, Andre Quina, Chris Moesel, Dylan Hall, Carlton Duffett, Kudakwashe Dube, Thomas Gallagher, and Scott McLachlan. Synthea: An approach, method, and software mechanism for generating synthetic patients and the synthetic electronic health care record. *J. Am. Medical Informatics Assoc.*, 25(3):230–238, 2018. doi: 10.1093/jamia/ocx079. URL https://doi.org/10.1093/jamia/ocx079.

[31] Anna L. Buczak, Steven M. Babin, and Linda J. Moniz. Data-driven approach for creating synthetic electronic medical records. *BMC Medical Informatics Decis. Mak.*, 10:59, 2010. doi: 10.1186/1472-6947-10-59. URL https://doi.org/10.1186/1472-6947-10-59.

[32] Scott McLachlan, Kudakwashe Dube, and Thomas Gallagher. Using the CareMap with Health Incidents Statistics for Generating the Realistic Synthetic Electronic Healthcare Record. In *2016 IEEE International Conference on Healthcare Informatics, ICHI 2016, Chicago, IL, USA, October 4-7, 2016*, pages 439–448. IEEE Computer Society, 2016. doi: 10.1109/ICHI.2016.83. URL https://doi.org/10.1109/ICHI.2016.83.

[33] Yubin Park, Joydeep Ghosh, and Mallikarjun Shankar. Perturbed Gibbs Samplers for Generating Large-Scale Privacy-Safe Synthetic Health Data. In *IEEE International Conference on Healthcare Informatics, ICHI 2013, 9-11 September, 2013, Philadelphia,*

PA, USA, pages 493–498. IEEE Computer Society, 2013. doi: 10.1109/ICHI.2013.76. URL https://doi.org/10.1109/ICHI.2013.76.

[34] Edward Choi, Siddharth Biswal, Bradley A. Malin, Jon Duke, Walter F. Stewart, and Jimeng Sun. Generating Multi-label Discrete Patient Records using Generative Adversarial Networks. In Finale Doshi-Velez, Jim Fackler, David C. Kale, Rajesh Ranganath, Byron C. Wallace, and Jenna Wiens, editors, *Proceedings of the Machine Learning for Health Care Conference, MLHC 2017, Boston, Massachusetts, USA, 18-19 August 2017*, volume 68 of *Proceedings of Machine Learning Research*, pages 286–305. PMLR, 2017. URL http://proceedings.mlr.press/v68/choi17a.html.

[35] Sankar K. Pal and Sushmita Mitra. Multilayer perceptron, fuzzy sets, and classification. *IEEE Trans. Neural Networks*, 3(5):683–697, 1992. doi: 10.1109/72.159058. URL https://doi.org/10.1109/72.159058.

[36] Augustus Odena, Jacob Buckman, Catherine Olsson, Tom B. Brown, Christopher Olah, Colin Raffel, and Ian J. Goodfellow. Is Generator Conditioning Causally Related to GAN Performance? In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 3846–3855. PMLR, 2018. URL http://proceedings.mlr.press/v80/odena18a.html.

[37] Samarth Sinha, Zhengli Zhao, Anirudh Goyal, Colin Raffel, and Augustus Odena. Top-k Training of GANs: Improving GAN Performance by Throwing Away Bad Samples. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, De-*

cember 6-12, 2020, virtual, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/a851bd0d418b13310dd1e5e3ac7318ab-Abstract.html.

[38] Zhengli Zhao, Zizhao Zhang, Ting Chen, Sameer Singh, and Han Zhang. Image Augmentations for GAN Training. *CoRR*, abs/2006.02595, 2020. URL https://arxiv.org/abs/2006.02595.

[39] Amirsina Torfi. *Privacy-Preserving Synthetic Medical Data Generation with Deep Learning*. PhD thesis, Virginia Tech, Blacksburg, VA, 2020. URL http://hdl.handle.net/10919/99856.

[40] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew P. Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 105–114. IEEE Computer Society, 2017. doi: 10.1109/CVPR.2017.19. URL https://doi.org/10.1109/CVPR.2017.19.

[41] Jiaxian Guo, Sidi Lu, Han Cai, Weinan Zhang, Yong Yu, and Jun Wang. Long Text Generation via Adversarial Training with Leaked Information. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 5141–5148. AAAI Press, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16360.

[42] Cristóbal Esteban, Stephanie L. Hyland, and Gunnar Rätsch. Real-valued (Medical)

Time Series Generation with Recurrent Conditional GANs. *CoRR*, abs/1706.02633, 2017. URL http://arxiv.org/abs/1706.02633.

[43] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 2242–2251. IEEE Computer Society, 2017. doi: 10.1109/ICCV.2017.244. URL https://doi.org/10.1109/ICCV.2017.244.

[44] Mehdi Mirza and Simon Osindero. Conditional Generative Adversarial Nets. *CoRR*, abs/1411.1784, 2014. URL http://arxiv.org/abs/1411.1784.

[45] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 4401–4410. Computer Vision Foundation / IEEE, 2019. doi: 10.1109/CVPR.2019.00453. URL http://openaccess.thecvf.com/content_CVPR_2019/html/Karras_A_Style-Based_Generator_Architecture_for_Generative_Adversarial_Networks_CVPR_2019_paper.html.

[46] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-Image Translation with Conditional Adversarial Networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 5967–5976. IEEE Computer Society, 2017. doi: 10.1109/CVPR.2017.632. URL https://doi.org/10.1109/CVPR.2017.632.

[47] Lucas Theis, Aäron van den Oord, and Matthias Bethge. A note on the evaluation of generative models. In Yoshua Bengio and Yann LeCun, editors, *4th International*

Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings, 2016. URL http://arxiv.org/abs/1511.01844.

[48] Holly E. Gerhard, Felix A. Wichmann, and Matthias Bethge. How Sensitive Is the Human Visual System to the Local Statistics of Natural Images? *PLoS Comput. Biol.*, 9(1), 2013. doi: 10.1371/journal.pcbi.1002873. URL https://doi.org/10.1371/journal.pcbi.1002873.

[49] Paulina Grnarova, Kfir Y. Levy, Aurélien Lucchi, Nathanaël Perraudin, Ian Goodfellow, Thomas Hofmann, and Andreas Krause. A Domain Agnostic Measure for Monitoring and Evaluating GANs. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 12069–12079, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/692baebec3bb4b53d7ebc3b9fabac31b-Abstract.html.

[50] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 248–255. IEEE Computer Society, 2009. doi: 10.1109/CVPR.2009.5206848. URL https://doi.org/10.1109/CVPR.2009.5206848.

[51] Chris Donahue, Zachary C. Lipton, Akshay Balsubramani, and Julian J. McAuley. Semantically Decomposing the Latent Spaces of Generative Adversarial Networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=S1nQvfgA-.

[52] Yann LeCun and Fu Jie Huang. Loss Functions for Discriminative Training of Energy-Based Models. In Robert G. Cowell and Zoubin Ghahramani, editors, *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics, AISTATS 2005, Bridgetown, Barbados, January 6-8, 2005*. Society for Artificial Intelligence and Statistics, 2005. URL http://www.gatsby.ucl.ac.uk/aistats/fullpapers/207.pdf.

[53] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a Similarity Metric Discriminatively, with Application to Face Verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), 20-26 June 2005, San Diego, CA, USA*, pages 539–546. IEEE Computer Society, 2005. doi: 10.1109/CVPR.2005.202. URL https://doi.org/10.1109/CVPR.2005.202.

[54] Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet. Are GANs Created Equal? A Large-Scale Study. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 698–707, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/e46de7e1bcaaced9a54f1e9d0d2f800d-Abstract.html.

[55] 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. doi: 10.1109/5.726791. URL https://doi.org/10.1109/5.726791.

[56] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *CoRR*, abs/1708.07747, 2017. URL http://arxiv.org/abs/1708.07747.

[57] Alex Krizhevsky. Learning Multiple Layers of Features from Tiny Images. Master's

thesis, University of Tronto, Toronto, Canada, 2009. URL https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

[58] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In Francis R. Bach and David M. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 448–456. JMLR.org, 2015. URL http://proceedings.mlr.press/v37/ioffe15.html.

[59] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier Nonlinearities Improve Neural Network Acoustic Models. In *ICML Workshop on Deep Learning for Audio, Speech and Language Processing, Atlanta, GA*, 2013. URL http://ai.stanford.edu/~amaas/papers/relu_hybrid_icml2013_final.pdf.

[60] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral Normalization for Generative Adversarial Networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=B1QRgziT-.

[61] Ishaan Gulrajani, Faruk Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville. Improved Training of Wasserstein GANs. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5767–5777, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/892c3b1c6dccd52936e27cbd0ff683d6-Abstract.html.

[62] Qiuyi Wu and Ernest Fokoue. Epileptic Seizure Recognition. UCI Machine Learning Repository, 2017. URL https://archive-beta.ics.uci.edu/ml/datasets/388.

[63] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL http://arxiv.org/abs/1312.6114.

[64] Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander J. Smola. A Kernel Method for the Two-Sample-Problem. In Bernhard Schölkopf, John C. Platt, and Thomas Hofmann, editors, *Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 4-7, 2006*, pages 513–520. MIT Press, 2006. URL https://proceedings.neurips.cc/paper/2006/hash/e9fb2eda3d9c55a0d89c98d6c54b5b3e-Abstract.html.

[65] Qiantong Xu, Gao Huang, Yang Yuan, Chuan Guo, Yu Sun, Felix Wu, and Kilian Q. Weinberger. An empirical study on evaluation metrics of generative adversarial networks. *CoRR*, abs/1806.07755, 2018. URL http://arxiv.org/abs/1806.07755.

[66] Alistair E.W. Johnson, Tom J Pollard, Lu Shen, Li-wei H Lehman, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. MIMIC-III, a freely accessible critical care database. *Scientific data*, 3:160035, 2016. doi: 10.1038/sdata.2016.35. URL https://doi.org/10.1038/sdata.2016.35.

[67] Alistair Johnson, Tom Pollard, and Roger Mark. The MIMIC-III Clinical Database, 2016. URL https://doi.org/10.13026/C2XW26.

[68] National Center for Health Statistics (U.S.) and Council on Clinical Classifications. *The International Classification of Diseases, 9th Revision, Clinical Modification: ICD-*

9-CM. DHHS publication. U.S. Department of Health and Human Services, Public Health Service, Health Care Financing Administration, 1980. URL https://books.google.com/books?id=K0XpALD2VjsC.

[69] R. Devon Hjelm, Athul Paul Jacob, Tong Che, Kyunghyun Cho, and Yoshua Bengio. Boundary-Seeking Generative Adversarial Networks. *CoRR*, abs/1702.08431, 2017. URL http://arxiv.org/abs/1702.08431.

[70] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. IRGAN: A Minimax Game for Unifying Generative and Discriminative Information Retrieval Models. In Noriko Kando, Tetsuya Sakai, Hideo Joho, Hang Li, Arjen P. de Vries, and Ryen W. White, editors, *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017*, pages 515–524. ACM, 2017. doi: 10.1145/3077136.3080786. URL https://doi.org/10.1145/3077136.3080786.

[71] Junbo Jake Zhao, Yoon Kim, Kelly Zhang, Alexander M. Rush, and Yann LeCun. Adversarially Regularized Autoencoders. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 5897–5906. PMLR, 2018. URL http://proceedings.mlr.press/v80/zhao18b.html.

[72] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient. In Satinder P. Singh and Shaul Markovitch, editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 2852–2858. AAAI Press, 2017. URL http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14344.

[73] Yann LeCun and Fu Jie Huang. Loss Functions for Discriminative Training of Energy-Based Models. In Robert G. Cowell and Zoubin Ghahramani, editors, *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics, AISTATS 2005, Bridgetown, Barbados, January 6-8, 2005*. Society for Artificial Intelligence and Statistics, 2005. URL http://www.gatsby.ucl.ac.uk/aistats/fullpapers/207.pdf.

[74] Amirsina Torfi, Seyed Mehdi Iranmanesh, Nasser M. Nasrabadi, and Jeremy M. Dawson. 3D Convolutional Neural Networks for Cross Audio-Visual Matching Recognition. *IEEE Access*, 5:22081–22091, 2017. doi: 10.1109/ACCESS.2017.2761539. URL https://doi.org/10.1109/ACCESS.2017.2761539.

[75] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic Differentiation in PyTorch. In *NIPS Autodiff Workshop*, 2017. URL https://openreview.net/forum?id=BJJsrmfCZ.

[76] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

[77] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning Deconvolution Network for Semantic Segmentation. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1520–1528. IEEE Computer Society, 2015. doi: 10.1109/ICCV.2015.178. URL https://doi.org/10.1109/ICCV.2015.178.

[78] Lilian Weng. From GAN to WGAN. *CoRR*, abs/1904.08994, 2019. URL http://arxiv.org/abs/1904.08994.

[79] Qingfeng Wang, Xuehai Zhou, Chao Wang, Zhiqin Liu, Jun Huang, Ying Zhou, Changlong Li, Hang Zhuang, and Jie-Zhi Cheng. WGAN-Based Synthetic Minority Over-Sampling Technique: Improving Semantic Fine-Grained Classification for Lung Nodules in CT Images. *IEEE Access*, 7:18450–18463, 2019. doi: 10.1109/ACCESS.2019.2896409. URL https://doi.org/10.1109/ACCESS.2019.2896409.

[80] Snehal Bhatia and Rozenn Dahyot. Using WGAN for Improving Imbalanced Classification Performance. In Edward Curry, Mark T. Keane, Adegboyega Ojo, and Dhaval Salwala, editors, *Proceedings for the 27th AIAI Irish Conference on Artificial Intelligence and Cognitive Science, Galway, Ireland, December 5-6, 2019*, volume 2563 of *CEUR Workshop Proceedings*, pages 365–375. CEUR-WS.org, 2019. URL http://ceur-ws.org/Vol-2563/aics_34.pdf.

[81] Cédric Villani. *Optimal Transport: Old and New*, volume 338. Springer Science & Business Media, 2008. doi: 10.1007/978-3-540-71050-9. URL https://doi.org/10.1007/978-3-540-71050-9.

[82] Karsten M. Borgwardt, Arthur Gretton, Malte J. Rasch, Hans-Peter Kriegel, Bernhard Schölkopf, and Alexander J. Smola. Integrating structured biological data by Kernel Maximum Mean Discrepancy. In *Proceedings 14th International Conference on Intelligent Systems for Molecular Biology 2006, Fortaleza, Brazil, August 6-10, 2006*, pages 49–57, 2006. doi: 10.1093/bioinformatics/btl242. URL https://doi.org/10.1093/bioinformatics/btl242.

[83] M H Zweig and G Campbell. Receiver-operating characteristic (ROC) plots: a fundamental evaluation tool in clinical medicine. *Clinical Chemistry*, 39(4):561–577, 04 1993. ISSN 0009-9147. doi: 10.1093/clinchem/39.4.561. URL https://doi.org/10.1093/clinchem/39.4.561.

[84] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership Inference Attacks Against Machine Learning Models. In *2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017*, pages 3–18. IEEE Computer Society, 2017. doi: 10.1109/SP.2017.41. URL https://doi.org/10.1109/SP.2017.41.

[85] Bradley A. Malin and Latanya Sweeney. How (not) to protect genomic data privacy in a distributed network: using trail re-identification to evaluate and design anonymity protection systems. *J. Biomed. Informatics*, 37(3):179–192, 2004. doi: 10.1016/j.jbi.2004.04.005. URL https://doi.org/10.1016/j.jbi.2004.04.005.

[86] Mrinal Kanti Baowaly, Chia-Ching Lin, Chao-Lin Liu, and Kuan-Ta Chen. Synthesizing electronic health records using improved generative adversarial networks. *J. Am. Medical Informatics Assoc.*, 26(3):228–241, 2019. doi: 10.1093/jamia/ocy142. URL https://doi.org/10.1093/jamia/ocy142.

[87] Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. Differentially Private Generative Adversarial Network. *CoRR*, abs/1802.06739, 2018. URL http://arxiv.org/abs/1802.06739.

[88] James Jordon, Jinsung Yoon, and Mihaela van der Schaar. PATE-GAN: Generating Synthetic Data with Differential Privacy Guarantees. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=S1zk9iRqF7.

[89] Leif E. Peterson. K-nearest neighbor. *Scholarpedia*, 4(2):1883, 2009. doi: 10.4249/scholarpedia.1883. URL https://doi.org/10.4249/scholarpedia.1883.

[90] Leo Breiman. Random Forests. *Mach. Learn.*, 45(1):5–32, 2001. doi: 10.1023/A:1010933404324. URL https://doi.org/10.1023/A:1010933404324.

[91] Tianqi Chen and Carlos Guestrin. XGBoost: A Scalable Tree Boosting System. In Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi, editors, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 785–794. ACM, 2016. doi: 10.1145/2939672.2939785. URL https://doi.org/10.1145/2939672.2939785.