Conditional Deep Generative Models for Generating Synthetic Electrocardiograms

Ramesh Upreti

Thesis submitted for the degree of
Master in Applied Computer and Information Technology (ACIT)
60 credits

Department of Computer Science
Faculty of Technology, Art and Design

OSLO METROPOLITAN UNIVERSITY

Spring 2023

Conditional Deep Generative Models for Generating Synthetic Electrocardiograms

Ramesh Upreti

Conditional Deep Generative Models for Generating Synthetic Electrocardiograms

http://www.oslomet.no/

Printed: Oslo Metropolitan University

Abstract

Using artificial intelligence (AI)-based diagnostic tools to assist healthcare professionals has increased significantly in recent years. However, AI is a heavily data-driven approach. Because of lack of data and especially imbalance nature of data, today's AI faces the most common issue of bias and overfitting. On the other hand, it is an undeniable fact that the lack of data in healthcare is one of the major issues. With the newer privacy rules, collecting and sharing data has become even more challenging. To mitigate this issue, the use of synthetic data emerged as an alternative solution. In this study, we aim to develop tools to generate synthetic data, and we have carried out extensive research on state-of-the-art models. Using electrocardiograms (ECGs) as a case study, we have then developed conditional deep generative models using generative adversarial networks (GANs) and denoising diffusion probabilistic models to generate 10 second long 12-lead ECGs based on a given input condition like desired heart rate. We trained the proposed models on two different datasets (PTB-XL and INT99+GENSUS). We evaluated the ECGs signals generated by normal and conditional models using different methods on both datasets, which demonstrated that the signals generated by GANS models are more realistic. The data generated by the best conditional GANs model yields FID distances of 3.36 and 4.73 on PTB-XL and INT99+GNSUS dataset, while the diffusion model yields FID distances of 16.65 and 10.97 respectively (note:lower FID is better). Similarly, the error of the machine learning (ML) classifier shows that the model has higher error in distinguishing real samples and fake samples generated by conditional GAN (accuracy: 79%, 87%) compared to the conditional diffusion model (accuracy: 92%, 92%) in both datasets. Power spectrum analysis shows that ECGs signal generated by conditional GANs model is almost identical to real signal strength, while the diffusion model signal strengths are lower than real which demonstrates that different waves are not correctly generated by diffusion model. Most importantly, ECGs parameters extracted using AI-model confirmed that both GANs and diffusion models are generating signals closer to the given conditional information. The extracted ECG parameters from the ECG signals, which are generated by conditional Generative Adversarial Networks (GANs), more accurately match the specified input conditions given in generative process. We have also verified the synthetic ECGs with an expert cardiologist. The cardiologist validated that the ECGs generated by our proposed models closely resemble the predetermined conditions. Among the two methodologies evaluated, the Generative Adversarial Network (GAN) models proved to be the superior approach. The data generated by our models is openly accessible to researchers. Furthermore, we have published all of our models as Python packages, which can be used for generating synthetic ECGs. In conclusion, our proposed models have demonstrated their capability to generate realistic ECG signals in accordance with specified conditions. Thus, they can be employed to create training datasets for the development of AI-based diagnostic tools, without incurring any privacy-related issues, among other benefits.

Acknowledgments

The use of synthetic data has been increased significantly and predicted that it will overshadow the real data in future. In such a time frame, I had the opportunity to conduct a deep study into the state-of-the-art methods for generating synthetic data. This thesis work has landed me on to the Ph.D. position at UiO before submitting it.

First and foremost, I would like to express my sincere gratitude to my supervisors Pål Halvorsen, Michael Riegler, and Vajira thambawita for their invaluable guidance, support, and encouragement throughout this thesis. Without their continuous feedback and support, this work would not have been possible. I would like to thank you one more time for this opportunity to work under your supervision.I would also like to thank our collaboration partners Jonas Isaksen and Jørgen Kanters from the University of Copenhagen. Their expertise knowledge in electrocardiograms plays an incredible role in improving and verifying the quality of synthetic data.

Similarly, I would like to use this moment to thank Oslo Metropolitan University for providing me with an opportunity to pursue my master's degree specialization in data science. My sincere gratitude goes to all my professors for their invaluable lectures, feedback, and encouragement. I would like to specially thank Professor Anis Yazidi and Pedro Lind for providing me an opportunity to collaborate in a research project while studying and guiding me to enhance my research skills. Without the use of dedicated GPU servers, the work was not possible, therefore, I would like to thank Simula Laboratory and University of Copenhagen for providing access to the servers.

Last but not least, I would like to thank my wife Jyoti Subedi Upreti for her incredible support and encouragement. I would like to remember my lovely daughter and parents for their love and support.

Acronyms

Following acronyms are used in this thesis work.

ECG - Electrocardiograms
AI - Artificial Intelligence
GDPR - General Data Protection Regulation
HIPPA - Health Insurance Portability and Accountability Act
GAN - Generative Adversarial Networks
R&D - Research and Development
VPN - Virtual Private Network
GPU - Graphics Processing Units
PC - Personal Computer
GB - GigaByte
WHO - World Health Organization
ML - Machine Learning
DL - Deep Learning
AE - AutoEncoder
VAE - Variational AutoEncoder
EEG - Electroencephalography
KL - Kullback–Leibler divergence
BCE - Binary Cross-Entropy
CNN - Convolutional Neural Network
LSTM - Long Short-Term Memory
TCN - Temporal Convolutional Networks
AdvP2P - Advance Pulse to Pulse
FID - Fréchet Inception Distance
IS - Inception Score

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Electrocardiograms (ECGs) are the measurement of cardio activity through an electrical signal. Measuring the ECG signal is an easy, fast, and non-invasive tool [35]. However, ECGs tools are potentially powerful tools which contain rich information about the healthiness of the heart. Indeed, ECGs signals are one of the key components used by doctors to diagnose and treat different kinds of diseases related to the heart [106]. Several research reports that ECGs signals can provide information related to arrhythmia[1], cardiac hypertrophy[2], myocardial ischemia[3], and many more diseases [31, 35].

Although ECGs signals are easy to capture and contain very potential information, ECGs signals are complex in nature [35]. Identifying the meaningful insides of ECGs signals is not an easy task. It requires an expert level of knowledge in the related field. On the other hand, several studies show that the lack of expert resources is the biggest problems worldwide [21, 61, 80]. One alternative solution is to train human resources, but it requires huge cost and time to train the human resources so that they can make clinical interpretations of ECGs signals. In addition to this, even though human experts are available, they are prone to interobserver variability in ECGs signals due to human errors (e.g., lack of concentration, focus, moods, etc.) [35].

To overcome above mention issues in the field of ECGs analysis, the use of artificial intelligence (AI) is rapidly emerging as an alternative solution in recent years [86]. Some of the early adoptions of AI models in the healthcare sector demonstrate the ability of AI in the medical sector. Indeed, artificial intelligence models outperform human experts in some medical tasks [7, 29]. For instance, in early 2020, the AI system developed by Google can identify breast cancer better than human experts [63]. In the domain of ECGs as well, studies show that AI-based analysis identifies ECG abnormalities with significantly higher accuracy than conventional algorithms and human experts [47, 50, 87].

However, it is an undeniable fact that today's artificial intelligence-based deep learning models are heavily dependent on the large volume of data [114]. Indeed, the lack of training data has a negative impact on the results of deep learning models. On the other hand, there is a huge scarcity of data in the medical field. There are two major reasons behind it. First, the collection of data requires a large amount of cost and time [91]. Second, in some cases; data might be available, however, it is very complicated to get access to and use those data due to privacy reasons [71]. Recent strong rules and laws imposed at the national and international

[1] https://www.webmd.com/heart-disease/atrial-fibrillation/heart-disease-abnormal-heart-rhythm

[2] https://www.cardiosecur.com/magazine/specialist-articles-on-the-heart/what-is-cardiac-hypertrophy

[3] https://www.mayoclinic.org/diseases-conditions/myocardial-ischemia/symptoms-causes/syc-20375417

level (GDRP rules introduced by the European Union[4], HIPPA rules imposed in the USA[5]) make it very difficult to use healthcare data related to the personal identity of patients.

To overcome the issue of data deficiency and improve the efficiency of deep learning models, generating synthetic data is one of the effective solutions [92]. However, it is very important that the generated data must capture the distribution of the original data in terms of quality and variations. In the domain of ECGs, traditionally mathematical equations were used to generate synthetic data. However, research shows that traditional methods require expert domain knowledge and generated data lose the standard in terms of quality and variation [22, 114]. Generative models enter the new era after the release of the deep learning-based model called the generative adversarial network (GAN) by Goodfellow in 2014 [33]. In fact, GAN has become one of the most widely used state-of-the-art methods for generating different types of synthetic data [17]. Similarly, after the release of denoising diffusion model [39] in 2021, diffusion models are also become state-of-the-art methods for generating images, and reached to the next level after the release of Stable Diffusion [76] and DALL-E and DALL-E 2 [74] model for generating images based on given input text.

The use of the generative adversarial network and diffusion models for generating realistic synthetic ECG data is getting higher attention from researchers in recent years. The current literature study shows that GAN and Diffusion models are used by researchers for generating ECG with varying numbers of channels and time intervals [1, 2, 41, 93, 106, 114]. However, there is a lack of research work in the direction of generating synthetic ECG data based on different conditions. Indeed, there are some work which take class labels (normal/abnormal) as input condition. However, in the domain of ECGs, ground truth of ECG, i.e., ECG parameters (such as Heart rate, PR interval, QRS duration, QT interval, etc.) are more prominent features than class labels. To the best of our knowledge, there is not a single work in the direction of using ECG ground truth (also called ECGs parameters) as conditional information. To build a better generalized model for ECGs analysis, it is important to have a sufficient number of ECG samples from diverse ground truth. Therefore, generating the ECG signal with multiple controllable conditions is the utmost for higher flexibility. To full fill this research gap, we are going to propose a novel deep learning based generative model with the capability of generating ECGs signals for given conditions.

1.2 Problem Statement

In the motivation section, we explained the different existing problems (most importantly lack of data with ground truth) and our proposed solution to overcome the mentioned problems and fulfill the research gaps. In this thesis, we carry out extensive research and investigation to figure out whether AI-based generative model architecture can take additional input of ground truth information as condition and enforce the model to generate ECGs signal equivalent to the given conditional information. Based on the motivation and the main goal, we can define that the main research question of this thesis work is as follows:

> *Can deep generative models generate realistic ECGs signals based on the given condition information?*

In the above research question, the term conditional information refers to the properties of ECGs. In the medical field, it is also called a biomarker. ECGs has different types of properties (biomarker) such as heart rate, PR-interval, QRS-duration, QT–interval, R-peaks, etc. In layman terms, the generative model generates ECGs at random, may generate ECGs with heart rate of 60 or 90, etc, but if we pass a heart rate of 80 to the model during generation, then the model

[4] https://gdpr-info.eu/

[5] https://www.hhs.gov/hipaa/index.html

also generates the ECGs signal with the heart rate of 80. To achieve the main goal of the defined research question, the following objectives are defined:

- **Objective-1:** Research and develop state-of-the-art deep learning based generative models (GAN and Diffusion) to generate realistic fake ECGs.
- **Objective-2:** Research and develop a suitable method to add a conditional parameter on proposed generative model so that model can generate ECGs of given condition.
- **Objective-3:** Research and develop different types of analysis methods for evaluating synthetic data.

1.3 Research Method

Artificial intelligence based generative models are one of the hot research fields in the domain of computer science which comes under the umbrella of scientific research. We follow the most widely used research framework proposed by ACM, which is divided into three paradigms: theory, abstraction, and design [23]. In this thesis work, we mostly follow the abstraction, which is also called modeling/prototyping paradigm. Abstraction (modeling) paradigm consist of following four stages: form a hypothesis, construct a model and make a prediction, design an experiment and collect data, and analyze results. This is a quantitative research. To overcome the limitation discussed in the previous motivation section, the following research steps will be followed.

- Define the main research question and define different objectives to support the main research question.
- Detail literature review will be conducted to figure out state-of-the-art work.
- Research and proposed a deep learning based generative model architecture.
- Design the experiment environment and conduct several experiments.
- Research and investigate different possible ways to evaluate the quality of work.
- Present research findings and result in a scientific way

1.4 Ethical Considerations

We make sure that all ethical considerations of the research work have been fully followed in this thesis work. We have worked with two different datasets, out of them, one is a private dataset owned by the University of Copenhagen. In the case of a private dataset, we have strictly followed the privacy rules of the university. The authors have also signed the confidentiality agreement for not sharing data and its misuse. By considering the privacy rules, we have conducted all our research on the university private server through VPN connection. To avoid the possible leakage of information, none of the public cloud-based services are used to track the experimental results. Similarly, to avoid the possible chances of adversarial attack, we are not publishing the trained model weights publicly. In addition to this, to ensure that none of the real data was exactly replicated in synthetic data, we performed the privacy test and made the data publicly available for the research community.

By considering the transparency and reproducing the same result, all the code of this thesis work is publicly available under the GitHub repository (https://github.com/upretiramesh/SyntheticECG/). In the case of a public dataset, we have also shared trained model weights and different analysis that has been performed on metadata and core data. The model trained in this thesis work donot have any bias and discrimination related to specific group of people. The anonymity of data is maintained by hiding identities of the participants. We further ensures that all the credibility of other work is fully maintained by referencing their work. The work is free of research misconduct and all the results are accurately presented. There are no any potential harms of this thesis work. The data generated from this thesis work can be used for further research and developing AI based diagnosis tools.

Last but not least, the project does not involve the use of human participants or animals. Therefore, ethical considerations related to human or animal subjects do not apply in this thesis work. We further confirmed that the authors have no financial or proprietary interests in any of the materials discussed in the thesis work. We declare no competing interests.

1.5 Scopes and Limitations

The study aims to produce high-quality synthetic ECG data that can be used for different purposes. In addition to quality, flexibility in generation is another major importance of study. In other words, this study allows researchers and doctors to generate the ECG signal for the desired conditions. Indeed, users can force the model to generate the ECG for given conditions. In the case of a private dataset, we have used the following seven parameters as conditions: *'VentricularRate', 'P_Duration', 'P_RInterval', 'QRSDuration', 'Q_TInterval', 'P_wave', 'PPeakAmp_II'* while in open source dataset following five features are used as conditions:*'QRS_Duration', 'QT_Interval', 'PR_Interval', 'AvgRRInterval','P_wave'*. The user can use the valid values for these features to use as conditions for generating ECG signals. Similarly, the study has proposed models based on two state-of-the-art methods called GAN and diffusion, which allows users to choose the model based on their requirements. To train the generative models, only normal (healthy) ECG signals are used. Therefore, the data generated by this study can be used for the analysis of normal ECG signals.

If the research community wants to use the data generated by this study or reproduce the same results presented in this study, we would like to mention some of the limitations of this work to be considered. First and most importantly, the study has considered only normal ECGs signals, therefore it cannot generate different types of ECGs related to different diseases. Similarly, it has used the fixed parameter as a condition which is mentioned above. Passing any other parameter may lead to useless generation. However, researchers can use the proposed model for training based on their requirement, that is, they can use different types of ECGs, and they can add or remove condition parameters. Similarly, all the models are trained on a dedicated server with a GPU of 32 GB, so training the model on normal PCs may not work or may take months to complete training.

1.6 Thesis Structure/Outline

The rest of the thesis work is organized in the following structure.

Chapter 2 - Background: In Section 2.1 we have described how artificial intelligence has been used in the healthcare sector. Similarly, Section 2.2 provides an overview of machine learning and its types. In addition to this, Section 2.3 provides an overview of different types of generative models and in-depth information about GAN, such as how it works, different types of loss functions, and different types of architecture. Last but not least, the detailed information of ECG and literature review is presented in Section 2.4.

Chapter 3 - Methodology: In the first two sections (3.1 and 3.2), we have explained about the dataset and the types of neural networks used in the proposed solutions. In section 3.3, we have proposed the architecture of GAN and its variants including discriminator and loss function. Similarly, diffusion-based model architecture is proposed in section 3.4. Passing conditional information to the model is one of the important aspects of this thesis work which we have explained in separate Section 3.5. Finally, the different types of evaluation methods are explained in Section 3.6.

Chapter 4 - Experiments and Results: how different experiments are conducted and evaluated are presented in this chapter. Section 4.1 describes experimental setup whereas Section 4.2 describes the model abbreviation used in this thesis work. Similarly, results from different models are evaluated in different methods which are explained in Sections 4.4 to 4.12.

Chapter 5 - Discussion and Future Work: Based on the experimental results presented in chapter 4, in this chapter, we discuss the different aspects of the result over different sections. The different kinds of solutions that we tried but not included as the main result are also discussed in this chapter. In addition to this, how this work can be extended in the future is also explained as a future work in section 5.2.

Chapter 6 - Conclusion: This chapter concludes the thesis work by highlighting the main findings of this study. Indeed, the research statement that we defined in the first chapter is achieved or not based on different objectives is verified in this last chapter.

Chapter 2

Background

In this chapter, we will explain all the necessary information related to this thesis work so that it builds the foundation to understand the necessity and importance of our work. We begin this chapter by providing how artificial intelligence has been used in the healthcare sector. To make it easier to understand technological sides, we will explain the core concept of machine learning and its different types. After this, we explain different types of generator models and their architecture. As our majority of work is in GANs, we will explain in depth about its concept and how it works. Similarly, the role of different loss functions with their pros and cons are explained so that readers can understand how the use of loss function can affect output. We introduce ECG, and different datasets used in the research domain. Finally, the literature review of related works will be summarized which will provide intuitive understanding of different approaches and their result.

2.1 Artificial Intelligence in HealthCare

Healthcare is the most important and one of the biggest sectors in the worldwide which has a direct relationship with the life and death of human beings. However, there are some major issues in the healthcare sector, for example, the lack of human experts: the shortage of 17.5 million health workers was reported in 2018 [65] and the shortage will increase to 18 million by 2030 according to WHO [58], human errors: Annual deaths of 44000 to 98000 patients are reported only in USA [52], and identifying hidden patterns from health care data (big data) is beyond human capabilities [16]. To resolve these issues and support healthcare persons, artificial intelligence has emerged as an alternative solution [81, 96, 98].

The use of artificial intelligence (AI) in healthcare started back around 1970 when a diagnosis tool called *MYCIN* was developed by a group of researchers from Stanford University to diagnose blood-borne bacterial infections [20]. The use of AI for *clinical decision system* were in used before couple of decades [20]. However, those decision systems have limited capabilities because they are based on an expert system which is a collection of if-then rules. With the recent advancement in the field of hardware development, availability of data, and powerful algorithms proposed by researchers, the field of artificial intelligence has reached the next level. Now, artificial intelligence can provide the decision in some tasks at the human experts' level or sometimes even better than experts without any human innervation. For example, a study conducted by Stanford University shows that AI beats human doctors in diagnosing cancers [29].

The proven success of artificial intelligence in image classification[1], image captioning[2],

[1] https://www.entrepreneur.com/article/283990

[2] https://www.dpreview.com/news/9384724203/microsoft-s-latest-computer-vision-technology-beats-humans-at-captioning-images

designing computer chips[3], game playing[4] and many more tasks have attracted the attention of researchers, doctors and healthcare organization to use AI in different healthcare sectors. In the healthcare sector, AI is most widely used in analyzing different types of images, e.g., computed tomography (CT), and magnetic resonance imaging (MRI). etc. Some of the promising results already prove the success of AI in medical image analysis [27, 29, 107]. However, AI is not only limited to image-related tasks in healthcare. It is also widely used in other tasks as well. For example, some medical tasks where AI is actively researching and using are clinical decision making, robot-assisted surgery, virtual nursing assistant, diagnostic, chat review and documentation, practice management, medical advice and triage, risk prediction and intervention, etc. [44, 57]. The use of AI assists healthcare professionals to make better decisions without wasting time.

The above paragraphs highlight the importance of artificial intelligence in the healthcare sector. AI has become an essential part of healthcare. Indeed, we cannot imagine the healthcare sector without the use of AI. If so, then what exactly is AI? The term *Artificial Intelligence* was coined by *John McCarthy* for the first time in 1956 during the Dartmouth college conference [62]. In addition to *John McCarthy*, *Alan Turing* is equally considered as father of artificial intelligence who wrote the article on *COMPUTING MACHINERY AND INTELLIGENCE* in 1950 [95]. In simple terms, artificial intelligence is a computer program written by a human which simulates the human intelligence in the machine so that machine itself becomes capable of mimicking human thinking and actions. There is no single standard definition of AI. The Oxford Dictionary defines artificial intelligence as:

> *"the theory and development of computer systems able to perform tasks normally requiring human intelligence, such as visual perception, speech recognition, decision-making, and translation between languages."*[72]

The term artificial intelligence (AI), machine learning (ML), and deep learning (DL) are normally used interchangeably in the field of technology. However, there are some key differences and strong relationships between these technologies. Out of these three terms, artificial intelligence is the umbrella term i.e. machine learning and deep learning are under the umbrella of artificial intelligence. Figure 2.1 demonstrates the relationship between AI, ML, and DL.

As shown in Figure 2.1, artificial intelligence has a broader range in addition to machine learning and deep learning. The nonoverlapping part of AI refers to a computer program (usually rule based logic, e.g., *if-then*) that adds human intelligence to the machine. In short, a set of predefined logic which does not depend on data. On the other hand, machine learning is a subset of artificial intelligence which learns and improves automatically from data and experiences without being explicitly programmed. ML is based on statistical learning algorithms which are suitable for the small to medium size of the dataset and for the small to medium level of difficult tasks. Therefore, ML algorithms are quick to learn and do not require dedicated hardware like GPU. On the other hand, deep learning is a subset of machine learning which brings machine learning to the next level. Deep learning mimics the human brain learning process using multiple layers of artificial neural networks. Deep learning is used for complicated tasks like work related to images, video, text, etc. DL can learn the features automatically and bring the machine's intelligence to the human level in some tasks. Compared to ML, DL applications are computationally expensive and require dedicated powerful computing resources. All deep learning algorithms are classified as machine learning and artificial intelligence, but not vice versa.

[3] https://www.nature.com/articles/d41586-021-01558-y

[4] https://www.bbc.com/news/technology-40042581

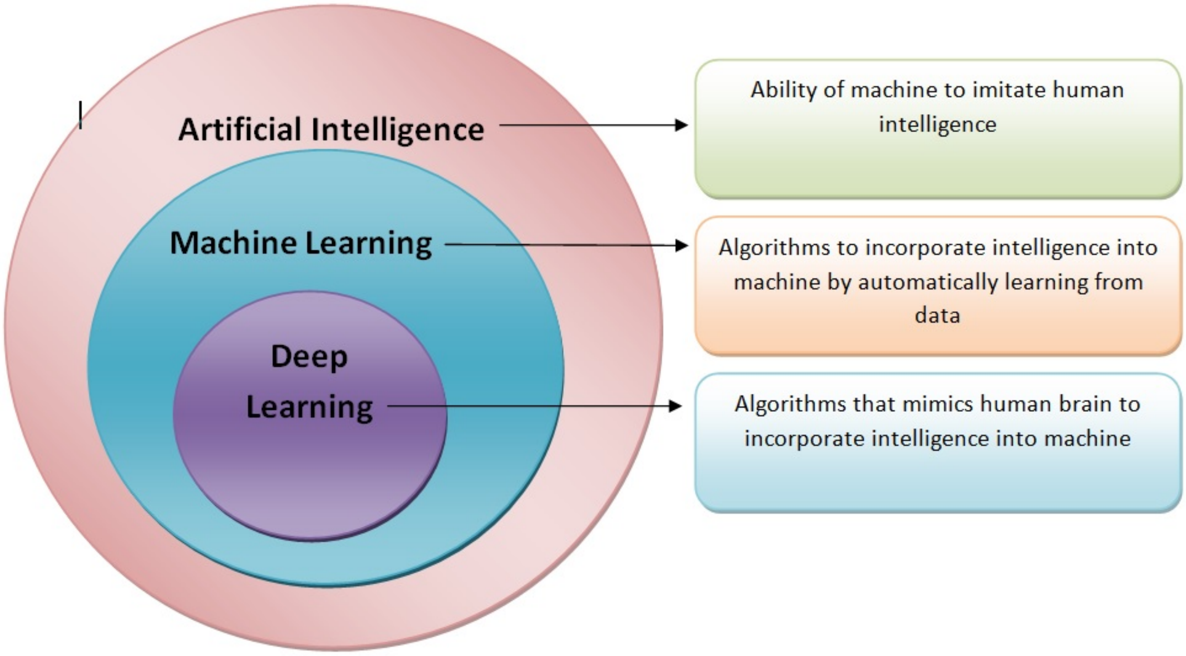

Artificial Intelligence
Machine Learning
Deep Learning
Ability of machine to imitate human intelligence
Algorithms to incorporate intelligence into machine by automatically learning from data
Algorithms that mimics human brain to incorporate intelligence into machine

Figure 2.1: Relationship of AI, ML and DL with definition (source : [4]).

2.2 Types of Machine Learning

Machine learning itself is a huge world that contains different types of models to solve different types of problems. All machine learning models are generally categorised into three subcategories. The purpose of this section is to give you an intuitive idea of these three subcategories of machine learning.

2.2.1 Supervised Learning

Supervised learning, which is also called supervised machine learning, is used in those problems where we have a labelled dataset. If we know the inputs and their corresponding correct output, then we can use supervised learning to train the model so that the model can learn the relationship between its input and output [83]. The training process starts with random weights and predicts the output. Based on the loss between the predicted output and the real output, the model updates its weight in a way so that the loss becomes minimal. The training of models happens iteratively until and unless it reaches to global minimum condition. Supervised learning is used in two problem domains called classification and regression. If the output is categorical, e.g True or False, Yes or No, 0 or 1, etc., then the classification algorithms are used to predict the output, whereas if the output is a continuous value (any number between $-\infty$ to $+\infty$) e.g the prediction of stock price, prediction of temperature, etc., then the regression models are used.

2.2.2 Unsupervised Learning

Unsupervised learning, also called unsupervised machine learning, is used in those problems where we do not have (or do not know) the output. The algorithms themselves find the hidden patterns or structures that exist in the training dataset without any human intervention. As the model is capable of identifying similarities and dissimilarities between data and revealing meaningful insights, these types of algorithms are also called knowledge discovery. In contrast to supervised learning, there is no corresponding output to compare the correctness of the model's prediction, therefore the training of unsupervised learning is more complicated and tricky compared to supervised learning [8]. Based on the nature of the problem, unsupervised learning is further classified into clustering, association, dimensionality reduction, and generative models.

2.2.3 Reinforcement Learning

Compared to supervised and unsupervised learning, reinforcement learning is an advanced concept in which we do not need any training dataset to train the model. In fact, reinforcement learning mimics the human learning process i.e learning by experience or learning from trial and errors [67]. In reinforcement learning, the agent takes an action based on the current environment that will yield the maximum reward. Reinforcement learning is widely used in training robots, autonomous vehicles, solving puzzles, etc. Reinforcement learning got tremendous attention from the research community after defeating the human experts of Go players in 2016 [85].

2.3 Generative Models

The solution that is going to propose in this thesis belongs to the category of unsupervised learning and subcategories of generative models. Therefore, in this section, you will get to know a brief introduction to generative models and some of the most popular types of generative models.

As we mentioned in the introduction section, one of the biggest issues in the field of machine learning is the deficiency of data. To solve this issue, generative models were introduced. As the name suggests, generative models are used for generating new dataset which looks like real data based on a probabilistic model. In mathematical terms, the main task of the generative model is to make/calculate $P(X)$ as high as $P(real)$ where $X, real$ refer to newly generated data and real data, respectively. Generative model is defined by David Foster [28] in his recent book called *Generative Deep Learning* as follows:

> *"A generative model describes how a dataset is generated, in terms of a probabilistic model. By sampling from this model, we are able to generate new data." [28]*

There are different types of generative models, however, we would like to explain some of the popular generative models in the following section.

2.3.1 AutoEncoder

AutoEncoder is a neural network-based model which leans to compress the data into lower-dimensional and reconstruct the new sample data from lower dimensional which are similar to input samples [10]. The model architecture of the autoencoder is presented in Figure 2.2. Autoencoder models are widely used in various applications including data generation such as denoising data, anomaly detection, clustering, recommendation system, dimensionality reduction, classification etc. Researchers have used this model for generating different types of data, for example, images, sound, ECG, EEG, etc. However, researchers reported that autoencoder models are data-specific [60, 70] i.e. can only produce data that looks like training data. In other words, it cannot understand the conceptual relationship between features. Similarly, the data generated by autoencoder model are lossy (encode and decode process lose output quality) [60]. Therefore, we cannot generate a different variety of high-quality data using an autoencoder.

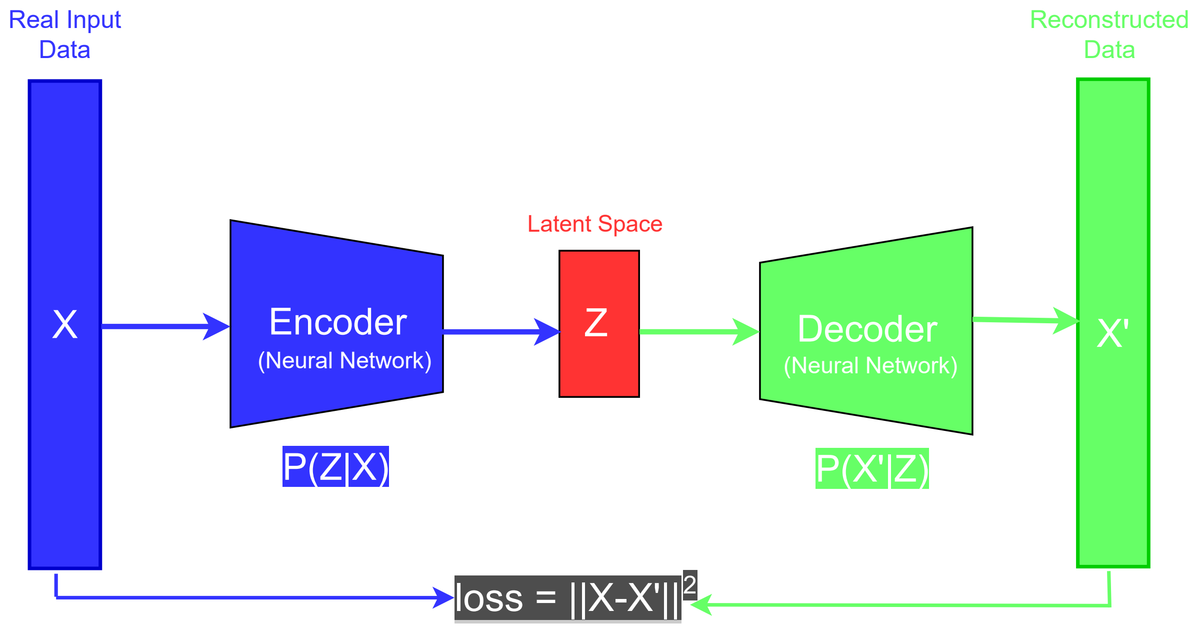


Figure 2.2: AutoEncoder model architecture. Encoder: can be any neural network which calculates probability of Z for given input, Latent Space (Z): lower dimension representation of input, Decoder: can be any neural network which reconstruct data from a latent space i.e calculate $P(X'|Z)$, Loss: reconstruct loss which make sure that input (X) data is ideally identical to generated data (X').

2.3.2 Variational AutoEncoder (VAE)

Variational autoencoder is an improvement over the autoencoder model [110]. AutoEncoder model encodes (maps) the input data into a fixed vector. Because of this nature, autoencoder models are limited to producing data which looks different than data that exists in training data. VAE overcomes this issue by encoding the input data into distribution. But, improving the model through distribution is very expensive, therefore VAE uses the trick of reparameterization to convert it into a single value. This gives the flexibility to the model to generate a variety of the same data, for example, a person with different skin tones. The model architecture is presented in Figure 2.3. Although VAE is an improvement over autoencoder, it still suffers from generating high-quality data and providing more flexibility (such as generating data based on condition, transforming one form of data into different forms adding the style of one data to another etc.) [42, 113]. Kuznetsov et al. [49] used VAE for the generation of ECG data, but while comparing the generated ECG data with other approaches (GAN), the quality was not good.

2.3.3 Denoising Diffusion Models

Diffusion models are another generative model which has received a lot of attention in recent times after the release of the research paper by OpenAI researchers in 2021 [24]. Diffusion models generate new sample data similar to the training dataset. The generation of the data happens in two steps called the forward diffusion process, in which the training data are destroyed by successively adding Gaussian noise until a defined number of time steps, while the reverse diffusion process is the backward process of generating new sample data by denoising the data at each step [100]. The researchers of [24] claimed that their proposed diffusion model can generate higher quality data and the training of the model is simple

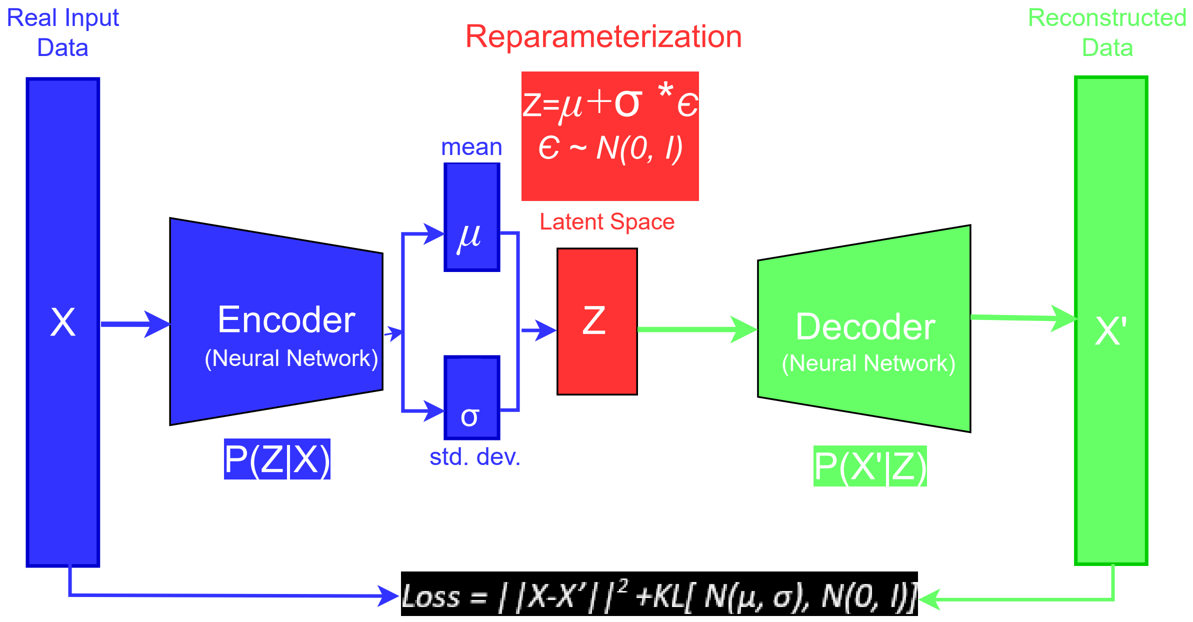


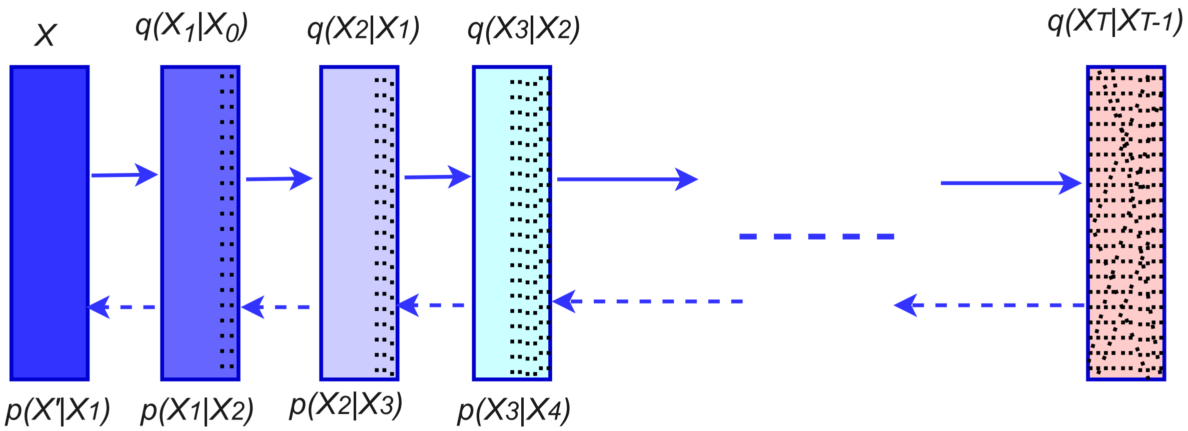
Figure 2.3: Variational autoencoder model architecture. Encoder: can be any neural network which output two dimensional outputs (mean and standard deviation), Latent Space (Z): calculated using a reparameterization trick which is required for backpropagation, Decoder: can be any neural network which reconstruct new data, Loss: in addition to reconstruction loss, *KL* (*Kullback–Leibler*) divergence is added which calculate the differences between two probability distribution.

compared to GAN. However, this model is quite new and has not been tested in many sectors.

2.3.4 GAN

Compared to the generative models discussed above, the generative adversarial networks (GAN) models are the most popular and widely used generative models to generate high-quality images [45], video [69, 99], audio [37, 105], time series data [15, 55], and in many more applications. The concept of GANs is considered one of the most beautiful ideas in the field of machine learning. In fact, the head of AI research at Facebook, *Jann Lecun*, mentioned that GAN is *"the most interesting idea in the last ten years in Machine Learning"* [68]. GANs was proposed by Ian J. Goodfellow and his colleagues for the first time in 2014 [33]. The generative adversarial network is made of three terms, Generative: which refers to generating a new set of data based on a probabilistic model, Adversarial: refers to competing for one against another which concept is used during the training of the model, and Network: which refers to any neural network-based algorithms. Combining all three terms, a generative adversarial network refers to a probabilistic model which learns how to map noise to a new set of realistic synthetic data which consist of two models called generator and discriminator. The generator's role is to generate a new set of realistic data from random noise (which is usually based on Gaussian or Uniform distribution for simplicity) and fool the discriminator while the discriminator's role is to distinguish whether the data is fake or real. Here, the generator and the discriminator are trained against each other in a competitive manner so that both models become better at their role. The model architecture of GANs is presented in Figure 2.5.

In contrast to autoencoder, VAE and diffusion model, instead of taking real samples as input, GANs architecture takes noise as input and learns the probability distribution to generate new synthetic data. The generator model is similar to the decoder block of the



Optimal Loss : prediction of noise added at each step of data. Mathematically looks as

$$L_{\text{simple}}(\theta) := \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2 \right]$$

Figure 2.4: Diffusion model architecture. Forward diffusion process: small amount of Gaussian noise is added to the input at each step until defined number of time steps (T), Reverse diffusion process: start from noise space i.e at T time step in a reverse order back to generate the new data sample by removing noise at each step, Loss: KL divergence is commonly used however researcher [39] reported that optimal result is gained by predicting the noise added at each time step. Dots refers to portion of noise exists in input sample and changes in color refers to changes appeared in data after added/removed noise.

autoencoder and VAE. On the other hand, the discriminator block somehow looks similar to the encoder block of the autoencoder, however, the encoder block convergence into latent space while the discriminator model is a binary classifier. As there are two separate models (Generator and Discriminator) in GANs, we need two different optimizers to tune those models. More details of the training process are explained in the section 2.3.4.1.

2.3.4.1 How GANs Works

The basic concept of GANs is explained in the previous section 2.3.4. In this section, we explain to you the working principle of the GAN model, in other words, the training process of GAN. The training process of GANs is different from the rest of the other models. In fact, training the GAN model is considered one of the complicated processes [56, 64]. For the sake of clarity, we explain the training process in detail.

Normally, the training process of GANs is divided into two parts. In the first part, the discriminator model is trained while the generator remains idle, which means that the generator only passes through the forward propagation and no backpropagation is done for the generator. There is a specific reason behind training the discriminator model first in the GAN training process which is that the generator loss itself depends on the prediction of the discriminator. Therefore, the discriminator first needs to determine how to distinguish between real and fake samples. In each epoch, the discriminator is trained both on a real sample and on fake samples. The loss is calculated from both samples (real and fake), which shows how correctly the discriminator can identify real samples as real and fake samples as fake. Based on both losses, the discriminator goes through the back-propagation process to update the model. The optimal discrimination goal is to correctly classify fake and real samples. In terms of loss value, the discriminator wants to maximize it as much as possible. The nitty-gritty detail of the

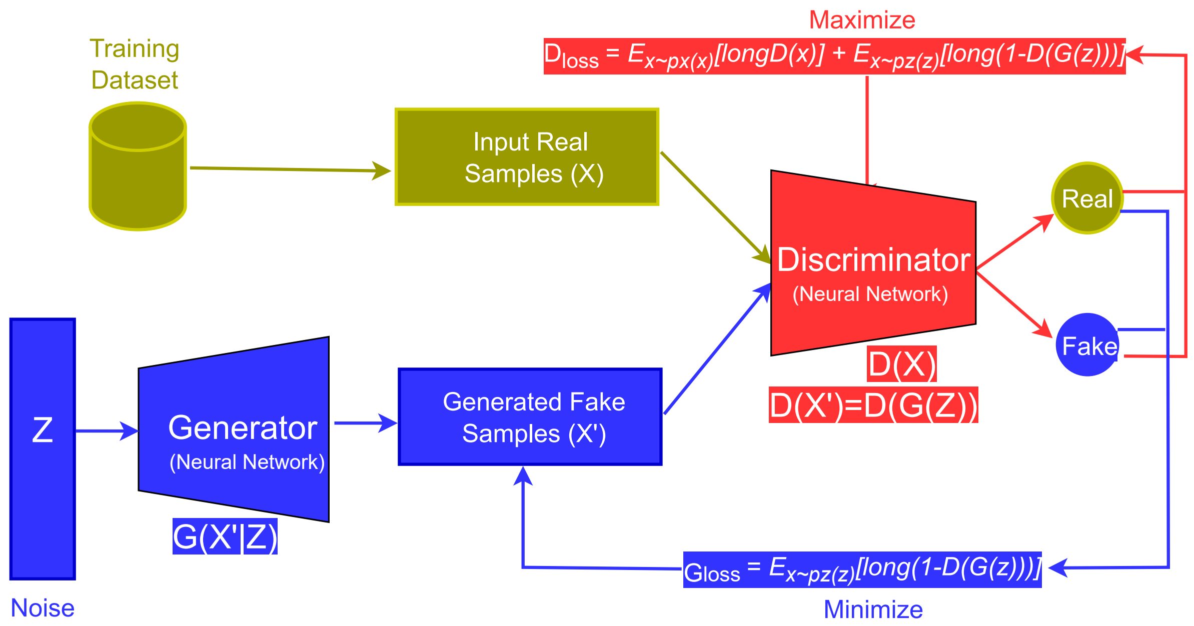


Figure 2.5: GANs model architecture. Generator: can be any neural network based model whose task is to generate fake sample data from random noise, Discriminator: can be any neural network based model which is train both on real sample and fake sample, and predict whether the inputs are real or fake, Discriminator Loss: try to maximize the loss so that real and fakes are correctly classified, Generator Loss: try to minimize the loss so that fake sample are also classified as real, Update: discriminator loss will update discriminator model and generator loss is used to update generator model.

loss function is explained in the below section 2.3.4.2.

The second part is vice versa of part one, i.e., the generator is trained on the second part while the discriminator remains idle. The generator generates the fake samples and passes them to the discriminator to classify them as real or fake. One important thing to understand at this point is that the intention of the generator is to generate realistic samples, in other words, to fool the discriminator so that the discriminator classifies the generated samples as real. Therefore, the loss is calculated in the assumption that the generated samples are real. The generator model is updated on the basis of this loss.

The convergence of GANs is another tricky or complicated decision in the GAN training process. After training the generator and discriminator on each epoch for many iterations, the generator becomes better and better at generating realistic samples, while the performance of the discriminator gets worse because it becomes harder for the discriminator to distinguish between real and fake samples. The ideal condition to stop the GAN model training process is when the discriminator accuracy reaches 50% of accuracy, which is the same as the prediction of the flipping coin. In reality, the discriminator accuracy may not reach 50% in most cases, therefore, stopping the GAN training when the accuracy reaches nearly 50% is good enough. Training the GAN model continuing after reaching the discriminator feedback almost at random conditions will have a negative impact on the quality of the generated samples [34].

2.3.4.2 BCE Loss Function

Like other optimization problems, the loss function plays an important role in the generative adversarial network in learning the distribution of real samples so that it can generate a similar distribution of data. Therefore, it is very important to understand the loss function of GANs and how it helps improve data generation. In this section, we explain the loss function proposed by Goodfellow in his paper [33] where the GAN was introduced.

$$E_x[log(D(x))] + E_z[log(1 - D(G(z))] \tag{2.1}$$

The loss function used in the original GANs paper is presented in the above equation 2.1. For the sake of clarity, let us explain to you the details of each term. The term $D(x)$ refers to the estimation of probability calculated by the discriminator for real sample data, E_x refers to the estimation value for all real samples, i.e. the mean probability score of all real samples classified as real, $D(G(z))$ refers to the probability estimation of generated fake samples as real calculated by discriminator and E_z refers to an expected value for all generated fake samples. This is the same loss function used by the generator and discriminator. The generator tries to minimize this function while the discriminator tries to maximize it. Therefore, it is also called the minmax loss function. As there are two models in GANs, the standard BCE loss function can be divided into two parts called discriminator loss and generator loss.

As we explained before, the discriminator is trained on both real samples and fake samples so that it can correctly classify real samples as real and fake samples as fake. Therefore, the discriminator will be penalized if it misclassifies real samples as fake and vice versa. The above loss function is used as it is in the case of discriminator i.e Discriminator loss = $E_x[log(D(x))] + E_z[log(1 - D(G(z))]$. The discriminator tries to maximize $E_z[log(1 - D(G(z))]$ this part so that the discriminator label the fakes samples as fake. To maximize $E_z[log(1 - D(G(z))]$ this function, the estimation value of $D(G(Z))$ needs to be zero or nearly zero. We know that the value of log(1) is 0. Therefore, maximizing this function will tend to lower the loss value.

On the other hand, the generator model is only trained on fake samples. There is no any role of real samples. Therefore, we ignore the loss from the real samples in the above standard loss function. The generator function becomes like this generation loss = $E_z[log(1 - D(G(z))]$. The generator tries to minimize this function so that it succeeded in the optimal goal of fooling the discriminator, which means that the discriminator will classify the fake sample as real. To minimize $E_z[log(1 - D(G(z))]$ part, $D(G(z))$ needs to output the value as 1 or near 1. If the generator is able to fool the discriminator, the generator loss gets rewarded, otherwise it will be penalized.

The implementation of the above standard GANs loss function can be achieved using the binary cross-entropy (BCE)[5] loss, which is rewarded for correct classification and penalized for misclassification. The formula of BCE loss for a single entry is presented in the below equation 2.2.

$$z = -y * log(\hat{y} + (1 - y) * log(1 - \hat{y}) \tag{2.2}$$

Let us first derive the discriminator loss from the BCE loss. As we know, a discriminator is trained on both real and fake samples where real samples are labelled as 1 and fake samples are labelled as 0. Applying BCE loss to the real sample, i.e., y = 1 gives the following equation 2.3:

$$\begin{aligned} loss_x &= y * log(\hat{y}) + (1 - y) * log(1 - \hat{y}) \\ &= 1 * log(\hat{y}) + (1 - 1) * log(1 - \hat{y}) \\ &= log(\hat{y}) \\ &= log(D(x)) \end{aligned} \tag{2.3}$$

[5] $https : //en.wikipedia.org/wiki/Cross_entropy$

Similarly, applying BCE loss on generated fake sample i.e y=0 gives following equation 2.4,

$$\begin{aligned}loss_z &= y * log(\hat{y}) + (1-y) * log(1-\hat{y}) \\ &= 0 * log(\hat{y}) + (1-0) * log(1-\hat{y}) \\ &= log(1-\hat{y}) \\ &= log(1-D(G(z)))\end{aligned} \tag{2.4}$$

After combining both losses, we get the same loss function proposed by Goodfellow i.e $loss = loss_x + loss_z = log(D(x)) + log(1-D(G(z)))$. This is a loss for single record, if we have multiples records, than E_x and E_z needs to be calculate to get a single loss value. Therefore, the loss becomes $E_x[log(D(x))] + E_z[log(1-D(G(z))]$ which is same as equation 2.1.

2.3.4.3 Problems With BCE Loss in GANs

Since the GAN model was introduced by Goodfellow, BCE loss is used by many researchers in different kinds of problems starting with the loss function [71, 114]. However, the researchers realized some of the major issues with BCE loss and reported about it in their article [6, 11, 94]. In this section, we will explain to you some of the major issues of BCE loss while using GAN architecture.

The purpose of GANs is to generate a large variety of samples that exist in the training dataset. For example, the GAN model train on numbers (0-9) should generate 10 different numbers. However, in reality, in most cases, the GAN model generates only limited numbers repetitively. This situation is called mode collapse in GANs [26]. Normally, real-life data are multimodal; in other words, the training data belong to multiple distributions of data. The generator is supposed to lean all the distribution that exists in the dataset. As we know, the generator's intention is to fool the discriminator. If the generator figures out a particular data generated by it can fool the discriminator, then the generator tries to generate a similar distribution of data in the next round, which will end the model to generate only one or a few distributions of data[6]. The discriminator never managed to learn how to break out of such a trap[7].

Another major issue of BCE loss in GANs is the problem of vanishing gradients. This problem occurs in GANs when the discriminator model is too good at distinguishing real and fake samples. When the discriminator distinguishes the real and fake samples with a very high probability, the loss declines toward zero. In such a case, the generator does not get constructive/informative feedback from the discriminator to improve the generation. This situation will lead to a vanishing gradient problem [101]. In the GAN architecture, the discriminator may become strong because the role of the discriminator is pretty simple compared to the generator. The discriminator's job is to output the value between 0 and 1, while the generator needs to learn to generate new data samples (same as it is easier to view the images in the museum than to create those masterpiece images by itself). In the starting of training, it is hard for the discriminator to distinguish between real and fake samples, so the generator gets more informative feedback from the discriminator. But as training continues, the discriminator becomes better and better quickly and could become 100% good at its job, creating the situation of vanishing gradients[8].

[6] https://www.geeksforgeeks.org/modal-collapse-in-gans/

[7] https://developers.google.com/machine-learning/gan/problems

[8] https://www.coursera.org/lecture/build-basic-generative-adversarial-networks-gans/problem-with-bce-loss-DzTcN

2.3.4.4 Optimization of GANs

Traditionally, BCE loss is used in training the GAN model which was responsible for two of the major problems which we explained in the previous section. To generate a variety of data and faster convergence (avoid vanishing gradient) of the model, it was essential to come up with a new solution. To overcome these issues, finally, Arjovsky and his team proposed a new loss function called *Wasserstein Loss* (also called W-loss in short) in 2017 [6]. The authors of [6] realized two major weaknesses in BCE loss which were responsible for the above-discussed problems. First, the output value of the discriminator is bounded to be between 0 and 1 because it uses the sigmoid activation function at the last layer and second, the discriminator needs to distinguish between real and fake samples. To avoid those weaknesses, the authors proposed a simple and efficient loss function. The loss function is presented in the equation 2.5.

$$E_x[D(x)] - E_z[D(G(z))] \tag{2.5}$$

Compared to BCE loss, the discriminator in Wasserstein GANs outputs a single real number value instead of a bounded value between 0 and 1 because it removed the sigmoid activation function at the layer and uses only a linear function. Similarly, the role of the discriminator is also changed because it no longer needs to distinguish between real and fake samples. Therefore, the discriminator of W-loss is also called a critic. In W-loss, the discriminator tries to maximize the distance between the real sample output and fake sample outputs, which means the discriminator maximizes the loss function presented in the equation 2.5. Maximizing the distance between two estimations tries to push the distribution of two samples as far as possible. On the other hand, the generator wants to minimize the distance of equation 2.5 as much as possible so that the discriminator realizes that the fake data distribution is close enough to real data distribution. As the output of the discriminator is not bounded, the discriminator always provides useful feedback for the generator to improve. Therefore, using w-loss mitigates the vanishing gradient problem which ultimately resolves the mode collapse problem.

The Wasserstein loss formula is derived from earth mover distance (EMD)[9] (also called EM distance) which calculates the distance between two probability distributions. The Wasserstein loss is considered valid when it meets the One Lipschitz continuity (1-L continuity)[10]. The 1-L continuity in discriminator confirms that the changes in the discriminator should be within the range of the slope of -1 to 1. This is essential in the case of a discriminator. As we discussed above, if the discriminator is too perfect, the generator gets very minimal gradient information to improve. To avoid such situations, 1-L continuity forcefully limits the gradient of the discriminator so that discriminator becomes worse however it provides much more information for the generator to improve. We recommend the reader to check the footnote links to know more about earth mover distance and Lipschitz continuity.

The efficient way of achieving the 1-L continuity is the open research topic. However, two of the methods are widely used in GANs. The first one is called gradient clipping. This is the concept used by the authors who proposed the Wasserstein GAN [6]. The authors mention that gradient clipping is a terrible way to achieve 1-L continuity; however, the authors used this concept because of its simplicity. It limits the gradient of critic within the range of $-c$ to c where c is a small constant value, e.g. $c = 0.01$ so that it makes it harder for the critic to reach its optimal condition.

The Wasserstein loss with gradient clipping helps to achieve stable training of GANs, however, sometimes models fail to converge or generate only low-quality samples. The authors of [36] argued that this is due to gradient clipping. To over this problem, Gulrajani and his team [36] proposed a new technique called gradient penalty. The gradient penalty maintains the 1-

[9] $https://en.wikipedia.org/wiki/Earth\_mover's\_distance$

[10] $https://en.wikipedia.org/wiki/Lipschitz\_continuity\#::text = In\%20the\%20theory\%20of\%20differential, the\%20Banach\%20fi$

L continuity of critic by penalizing the norm of a gradient with respect to its input [36]. The authors claimed that this method outperforms the gradient clipping method and makes the training of GANs more stable. The wasserstein loss with gradient penalty is a widely used method in different types of GANs solutions after its release.

In addition to the above methods, researchers have also proposed other methods as well in recent years to make the GANs training more stable. We briefly mention those methods and recommend that the reader check the reference paper for in-depth understanding. In 2018, Miyato and his team proposed a new method called *spectral normalization* to make GANs training stable. Similarly, researchers, Lin and Qiu proposed a new method called *boundedness and continuity (BC)* to satisfy a Lipschitz constraint [59] in 2020. The authors claimed that their proposed BC method not only performs better but is also computationally efficient as compared to gradient penalty and spectral normalization [59]. Similarly, *adaptive weighted loss* also called *aw-loss* is one of the recent loss function proposed by Zadorozhnyy and his colleagues in 2021 [108]. The proposed discriminator loss function is a new concept compared to previous methods which adopt the new weight of the discriminator based on the discriminator's weights while training on real samples and discriminator's weights while training on fake samples. In short, the new gradients will be the weighted sum of gradients from both real and fake losses. The author claimed that the proposed loss function can be used in any GAN architecture and improves the results significantly as compared to other methods [108].

2.3.4.5 Major Types of GAN Architecture

The first GAN model proposed by Goodfellow in 2014 was unable to produce high-quality synthetic samples. However, the concept of GANs got huge attention from researchers and big giants IT companies who helped to bring the GAN to the next level by extending the initial GANs framework. In a short period of time, different kinds of architecture were proposed that could generate high-quality data. In this section, we briefly describe some of the most popular types of GANs architectures.

Deep Convolution GAN (DCGAN): The concept of DCGAN is pretty simple. Instead of using vanilla neural networks (which was used by Goodfellow [33]), the generator and discriminator both use convolution layers. The generator uses transpose convolution to upsamples the data, while the discriminator uses strided convolution to down samples the data at each layer. This concept was proposed by researchers from Facebook and Indico in 2015 [73]. After the release of this paper, most of the papers used the convolution layer in GAN. To know more about the model, we recommend the reader to check reference [73].

Conditional GANs (CGAN): Although GANs can generate new synthetic data, it has no control over what to generate. In other words, the training dataset is usually multimodal, but the generator and discriminator do not have any idea about what the modal is going to generate and discriminate. To overcome this issue, Mirza and Osindero proposed the idea of conditional generative adversarial networks [66]. In conditional GANs, we can pass the condition on to the generator and discriminator so that it knows what it is supposed to generate and discriminate. Usually, conditions are labels of data. In CGAN, in addition to noise, the encoded label information is concatenated with noise and given to the generator as input. Similarly, real/fake sample data and associated encoded label information are concatenated and given to the discriminator as input.

Progressive GANs: The main purpose of the progressive GAN [46] is to improve the quality, stability, and variation of the generated samples. As the name suggests, the generator of progressive GANs starts generating a lower resolution image and adds more details in subsequent layers so that as the training progresses, the quality of images also progresses. The training of the model happens in step-by-step manner. For example, if the sample data is generated in six progressive steps, initially the model has only one layer, it first trains the model, calculates the loss, and updates the model, then it adds a second layer on top of the previous model, and repeats the same process until the final step, which means training, calculating loss, and model updates happen six times for each batch of data. The authors claimed that training the model in progressive steps helps to converge the model quickly and, at the same time, generate high quality data [46].

CycleGAN: CycleGAN [115] is another interesting idea in the family of GANs, which learns to translate one domain of data into another domain of data. The most common example of CycleGAN is inputting the image of a horse and getting the image of a zebra as an output and vice versa. As the model transforms the images from one domain to another, it needs to have two datasets belonging to two different domains. In the training process, two generators and two discriminators are required so that one pair of generators and discriminators learns to translate domain A to domain B while another pair of generators and discriminators perform the vice versa operation. To know more about CycleGAN architecture, we recommend that the reader check reference [115].

Super Resolution GAN (SRGAN): As the name suggests, super-resolution GAN is designed with the purpose of generating high-resolution images. Compared to the standard GAN model, instead of taking the noise as input, SRGAN downsampled the high-resolution images into lower resolution and passed them to the generator as input. On the other hand, the discriminator job is similar to the standard GAN model, it takes both real high-resolution images and fake generated images and distinguishes whether the images are high resolution or not. This idea is proposed by a team of researchers from Twitter [53]. The authors claimed that SRGAN can recover photorealistic textures from very low-resolution images. In addition to the adversarial loss in the generator, the SRGAN model also used content loss, which is commonly known as reconstruction loss. The author has used mean square error (MSE) to compute the context loss at a pixel level. For more information, we recommend the reader to check the reference [53].

InfoGAN: The full form of InfoGAN is information maximizing generative adversarial networks. This is a similar but more advanced concept of conditional GANs. In conditional GANs, we have control over what GANs is supposed to generate however GANs does not have control over many other features, for example, angle of images, the colour tone of face or hair, etc. InfoGAN provides this flexibility to generate new synthetics data by providing maximum information [18]. The authors claimed that InforGAN can produce an impressive result in generating images with different hairstyles, a person with or without eyeglasses, changing the emotion of a person, etc. The generator of InforGAN combines three different pieces of information (noise, categorical features - what to generate, continuous features - how it needs to look like) and takes it as input. Similarly, the discriminator not only distinguishes between real and fake, but also learns an interpretable representation of the generated images. For depth understanding, we recommend checking the reference [18].

BigGAN: GANs shows promising results in generating smaller size images with high quality. However, generating large-size images (i.e., high resolution) with high fidelity, which is also called high equality of images, is still a challenging task. Many of the researchers focused on changing the objective function or making the discriminator worse through the gradient penalty. In such a situation, BigGAN comes with the idea of changing the model design and the training process [13]. BigGAN is built on top of the self-Attention GANs [111] and used the simple concept of increasing batch size and scaling up the model, which ultimately helps to achieve high image quality. The authors have used different kinds of concepts like hinge loss, class conditional information, spectral normalization, orthogonal weight initialization, skip-z connection, truncation trick, updating discriminator more than the generator, and model weight updates using moving average, etc. We recommend the reader to read the official paper to know more about BigGAN [13].

2.4 Electrocardiograms (ECGs)

This thesis work is dedicated to the field of ECGs. Therefore, in this section, we are going to explain what an ECGs signal is, some common data sets in the ECG domain, and a review of the work related to ECGs generation.

2.4.1 The ECG Signal

In the domain of the heart, ECGs signals are one of the most important parameters to determine the correct assessment of how the heart is functioning. Indeed, ECGs signal provides huge amount of information related to the healthiness of a person's heart. ECGs signals are recorded using a non-invasive tool called electrode, which is placed on the chest area of skin and captures the electrical activity of heart [41]. Electrical activity is recorded in the form of waves which represent changes in height and depth of voltage based on heart functioning. Many researches show that there are noticeable differences in between the electrical waves (ECGs signal) of normal patients and patients having different kinds of heart diseases [9, 40, 75]. Therefore, ECGs signals are one of the most widely used diagnosis tools by doctors for the diagnosis and treatment of heart patients [106].

For a layman, ECGs signal looks like a normal wave with some ups and downs. However, for experts, ups and downs have their own meaning and contain lots of information related to the health status of patients. Therefore, it is very important to understand the life cycle of the ECG signal. Figure 2.6 represents one cardiac cycle of ECGs signal. In the Figure 2.6, you can see English alphabet letters P, Q, R, S, T and U which represent different kinds of waves exist in one cardiac cycle. As shown in Figure 2.6, P wave is the first wave in an ECG signal which represents atrial depolarization. Increase of the degree of P wave compared to normal wave or missing of P wave represent different kind of abnormalities of heart [51].

Similarly, the QRS interval, which is also called QRS complex, consists of Q, R and S waves. QRS interval indicates ventricular depolarization which is the most prominent feature of the ECG signal, where Q and S are down waves, and R is upward waves [75]. Cardiac hypertrophy may be detected through the abnormal patterns of QRS intervals. Ventricular repolarization of the heart is represented by T waves. The ST segment is another prominent feature of an ECGs, which indicates the time gap of ventricular depolarization and repolarization. The ST segment helps in identifying myocardial infarction or myocardial ischemia through a higher or smaller (or missing) value of ST segment [51]. The last wave is called U wave which is rarely visible in the ECG signal. Therefore, in many of ECGs cycle images U wave may not be included. The existence of U wave indicates the repolarization of the Purkinje fibers [51]. Understanding all these waves of the ECG cycle will helps to cross-verify the correctness of newly generated ECGs signals.

2.4.2 Commonly Used ECG Dataset

Different types of datasets are available in ECGs domain. To make data access easy, Physionet publishes most public ECGs datsets at the same location (*https://physionet.org/about/database/*). In this section, we have summarized some of the most commonly used datasets by the research community in the domain of ECGs generation with some details of the datasets. The information is summarized and presented in Table 2.1.

2.4.3 Generative ECGs Models: Review

Generation of ECGs signals has been in practice for a long time in the past. In this section, we are going to summarize different types of generative models proposed by researchers to generate synthetic ECGs data.

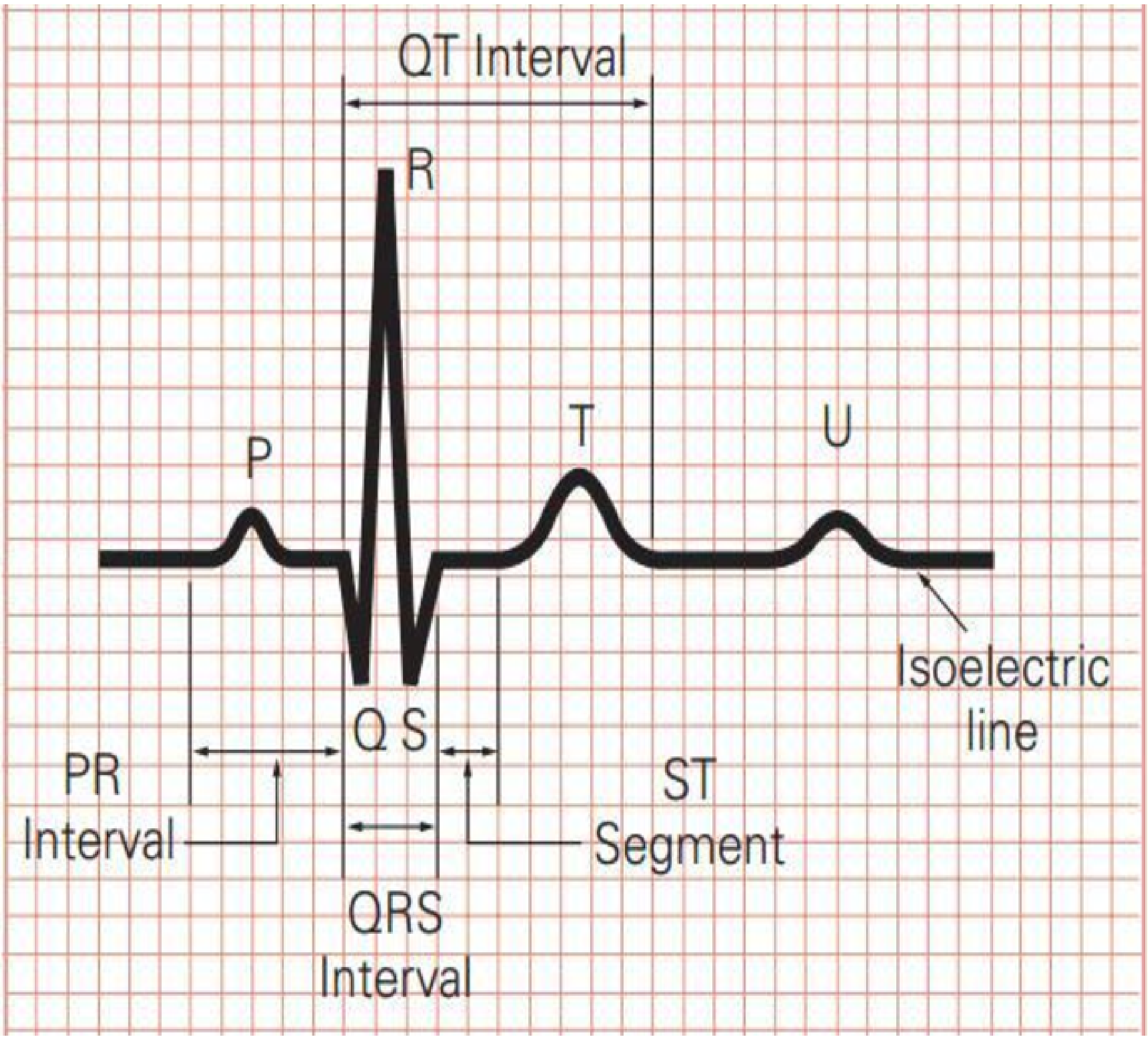


Figure 2.6: One cardiac cycle of ECGs signal [75], where the horizontal line represents time period and vertical line represents changes in electrical voltage level. Measurement of PR interval, QRS interval, ST segment, and QT interval length are four important characteristics in the diagnosis of patent's heart.

Name	References	Samples	Time Interval	leads	Availability
MIT-BIH	[41, 106, 114] [22, 31, 103] [14, 82, 84] [30, 32]	47	30-minutes	2	public
LUDB	[48]	200	10-seconds	12	public
PTB	[71]	549	N/A	15	public
PTB-XL	[5]	21,837	10-seconds	12	public
GESUS	[93]	8,939	10-seconds	8	private
Inter99	[93]	6667	10-seconds	8	private
RMN	[3]	661,509	0.5-seconds	1	private

Table 2.1: Some of the most commonly used ECGs dataset by research community with the detail of dataset name, reference, number of samples, time interval, number of leads and nature of availability.

Reference	Year	Types	Method	leads	Time
[88]	2001	Mathematical Equation	partial differential equations and ordinary differential equations	N/A	N/A
[78]	2004	Mathematical Equation	two set pair of Liénard equations	N/A	N/A
[19]	2006	Mathematical Equation	set of Gaussian functions with different width and height	12	N/A
[12]	2010	Mathematical Equation	partial differential equations	12	1-C
[109]	1996	Mathematical Equation	ordinary differential equation	N/A	N/A
[49]	2021	Variational Autoencoder	encoder: CNN, BN, ReLU decoder: upsample, CNN, ReLU	1	1-s
[32]	2020	GAN	inpired by DCGAN	1	216-p
[114]	2019	GAN	generator:LSTM discriminator:CNN	1	3120-p
[14]	2020	GAN	generator:LSTM discriminator:CNN minibatch discriminator	2	3120-p
[103]	2020	GAN	three different GAN models inspired by wavenet and DCGAN	1	varying
[93]	2021	GAN	GAN model inspired by U-NET	12	10-s

Table 2.2: Summary of generative ECG models with the details of reference, published year, type of generative models, main methods, number of channels and time interval. N/A refers to Not Available (mention). In Time column: C-cardiac cycle, s-seconds, p-number of points(length).

Using mathematical equations to generate the ECG signal was a common practice in the past. For example, Boulakia et al. [12] used partial differentiation to generate 12 lead ECGs signal, Sundnes et al. [88] used partial differential equation along with ordinary differential equations for modeling ECGs signals, set of Gaussian functions are used by Clifford et al. [19] to model segmented regions of ECGs signal, Lienard equations is used by Santos et al. [78] for modeling ECGs signal, Zbilut et al. [109] used three ordinary differential equation to generate P, QRS and T wave, etc. Although different types of mathematical equations were used for modeling ECGs signals, researchers reported that those traditional methods are limited in a number of ways [106]. For example, the generated signal distribution is not close enough to the original distribution, difficult to generate a high variation of the ECG signal, and not reliable for generating long -time interval signals [90, 93]. In addition to this, developing and modifying equations requires a level of mathematical knowledge of expertise and the knowledge of the expert domain [22]. The manual feature extraction process is another pitfall of the traditional approach [32].

In recent years, the use of deep learning based models is becoming popular for generating ECGs signals because you no longer need to have expert knowledge; at the same time, it can generate realistic fake samples. Kuznetsov et al. [49] proposed deep learning based variational autoencoder model for generating ECGs with a duration of one cardiac cycle. The authors have preprocessed the data and split the original data of 12-leads 10 second ECGs signal into separate 9 seconds signal where each signal has a length of 400. The encoder block consists of multiple blocks of convolution neural network (CNN)[11], batch normalization[12] and ReLU[13] activation layers while decoder block consists of multiple layers of upsamples, convolutions and ReLU activation layers. The size of the latent space is 25. The authors claimed that the train model yields the Maximum Mean Discrepancy (MDD) metric of 3.83 x 10 -3, which indicates that the generated ECG signals are of good quality. As the model only generates 1-lead ECGs with 1-second length, the model may not perform a good job of generating a heavy ECGs signal, that is, 12-leads ECGs with 10 seconds length.

Similarly, Golany et al. [32] presented another interesting work in the field of ECGs generation. The authors original purpose is to improve the ECG classification, but there was a lack of data. Therefore, to overcome the shortage of data, the authors proposed a GAN approach to generate synthetic ECGs data. The authors have taken inspiration from DCGAN architecture [73] to design the generator and discriminator model. The authors used binary cross entropy as the loss function. The generator generates 1-channel ECG with 216 points. The authors do not mention about quality of the generated ECG data in the results section; however, the authors show that ECGs classifiers yield better accuracy after adding synthetic ECGs data which indirectly indicates that generated samples are of good quality. The model is generating varying sample size of data (1x216); therefore, it might fail to generate a real scenario higher sample data (8x5000).

Zhu et al. [114] come off with the idea of using bidirectional LSTM[14] layers in the GAN architecture when most of the work were based on convolution layers. The authors used two layers of BiLSTM with generator which take 3120 noises as input and output the same size. The authors generated only one channel ECGs signals. The discriminator model is created using convolution layers. The authors argued that the generated output of ECGs signal is realistic. To conform the strength of the proposed GAN model, authors compared the output of the model with RNN-AE (recurrent neural network[15]-autoencoder) and RNN-VE (recurrent neural network - variational autoencoder) models. The comparison result yields that the proposed GAN based model yields a better result.

[11] https://en.wikipedia.org/wiki/Convolutional_neural_network

[12] https://en.wikipedia.org/wiki/Batch_normalization

[13] https://en.wikipedia.org/wiki/Rectifier_(neural_networks)

[14] https://en.wikipedia.org/wiki/Bidirectional_recurrent_neural_networks

[15] https://en.wikipedia.org/wiki/Recurrent_neural_network

Brophy [14] presented another recent work published in 2020. The authors argued that existing research work was limited in generating single channel ECGs data. Therefore, the authors proposed a GAN architecture to generate multivariate ECGs data. Indeed, the proposed GAN model generates 2-channel ECGs data with 3 cardiac cycles. The author extends [114] work where the generator has two layers of LSTM hidden layers followed by fully connected layers and discriminator has four layers of convolution layers with pooling layers. To avoid the mode collapse, the author used minibatch discriminator concept in the discriminator model. To confirm the quality of the generated data, the author has used Maximum Mean Discrepancy (MMD) and also introduced the new concept called multivariate Dynamic Time Warping (DTW), which measures the similarity across dependent signals. The author claimed that the generated ECGs signals are of good quality. The DTW score becomes minimal after 30 epoches, which also confirms the quality of the generated data.

Wulan et al. [103] proposed three GAN based models (called WaveNet-base, SpectroGAN, and WaveletGAN) to generate ECGs signals. The authors used μ-law companding transformation, short-term Fourier transform, and stationary wavelet transform as preprocessing steps in WaveNet, SpectroGAN, and WaveletGAN respectively. ECGs signals are generated using three different models that look promising. However, authors reported some issues in the models. For example, ECGs signal generated by WaveNet is not smooth as compared to SpectroGAN and WaveletGAN. Similarly, inspection of the output generated by SpectroGAN reveals that there is a lack of diversity in the output. In addition to this, the output generated by WaveNet is lacking in terms of quality and distribution. The authors have generated 1 lead ECGs signal with varying time intervals. The authors argued that one of the models can generate an ECGs signal of 20 seconds long.

Thambawita et al. [93] presented one of the most recent works in the domain of ECGs generation using GAN architectures. The authors proposed the GAN model inspired by U-Net [77] architecture, which is capable of generating 12 lead signals with the length of 10 seconds interval. The generator model of the GAN takes 8 x 5000 noise as input and generates the same size of data as output. The generator used six layers of down sample blocks. The output of each block is passed through Phase Shuffle layers and concatenated with the corresponding upsamples layers. The authors have compared the output of the proposed model with WaveGAN model [25] (used for generating audio) by modifying it. The authors showed that their proposed Pulse2Pulse GAN model outperforms the WaveNet model. This is one of promising works in the generation of realistic ECGs signals with higher number of channels.

2.5 Summary

In this chapter, the relevant background needed to understand this thesis work and the related work carried out by different researchers is explained. The explanation of the core concept of artificial intelligence and type of machine learning helps to understand the technological field. Similarly, use of AI in healthcare demonstrates that the use of AI in healthcare has been significantly increasing and yielding promising results. We explained different types of generative models with their architecture. Similarly, we explain how the Wasserstein loss overcomes the limitation of BCE loss. In addition to this, we explained the ECG and its components so that later you can understand what we are generating.

The literature review demonstrates that researchers have used different types of generative models to generate ECGs signals. The literature review shows that most of the previous work was limited to a smaller number of channels and for a short period of time. Similarly, some recent work shows that generative models are also used successfully for generating higher channels with a time interval of 10 seconds. However, we realized that most of the work does not consider passing conditional information as input. Most importantly, there exist some conditional models in recent work, however, they do not consider ground truth as conditional, instead use gender or class labels as conditional inputs. In the context of ECGs, whether we pass gender or class labels as conditions, the differences will be noticeable in the ground truth level. Therefore, passing the ground truth as conditions is a more promising approach. To the best of our knowledge, this is the first work in the domain of generative models which takes the ground truth as conditional information.

Chapter 3

Methodology

In this section, we are going to explain in detail about the research methods we have used in this thesis work to address the main research question. First of all, we have explained about what kind of neural network models are used in our methods, then we explain about the model architecture (how it looks like). For the sake of clarity (To make it easier for the reader to replicate the work), we have explained the model architecture and its components in detail including loss function. Finally, different types of evaluation techniques are discussed to verify the quality of the generated data.

3.1 Dataset

The main research question that we defined in our problem statement section is to generate the ECG based on the given condition. In the case of ECGs data, the condition could be information related to sex, age, types (class labels) of ECGs, etc. However, at depth level, the ground truth of the ECG, for example, heart rate, P-interval, QT-interval, QRS-duration, etc. are more useful information to pass as a condition. Therefore, it is very important for us to choose the dataset which contains these kinds of information in addition to ECGs. To the best of our knowledge, we find PTB-XL[1] is the most suitable dataset for our research. The PTB-XL dataset is suitable in several ways; for example, in addition to the ECG, it has provided information related to patients (such as age, sex, etc.), similarly, information related to the ECG (such as the class of each ECG). In addition to this, we also have the ground truth of this, which is provided by experts. Here, ground truth refers to features related to ECGs such as the P-interval, QT-interval, RR-interval, etc. which are used by doctors to determine the healthiness of people. Having access to the ground truth of this dataset is a great advantage for us so that we can use them as a condition as well as use them to compare with the generated samples. Furthermore, this is one of the large datasets publicly available (on: https://www.physionet.org/content/ptb-xl/1.0.2/) that contains 21837 records of 18885 patients. The PTB-XL dataset contains 10 second length ECGs records captured by 12 leads, which is another good side because in real life doctors use 12 lead ECGs signals for analysis. Most of the published dataset has one to 2 leads of ECGs signals.

While closely working on $PTB - XL$ dataset, doctors notice some of the odd patterns in the ECG signals. In ECGs domain, those types of odd patterns are referred to as horse and zebra. These types of signals increased complexity and make it very difficult to analyze the normal and abnormal ECGs. Therefore, doctors suggested that we train the model on a private dataset owned by the University of Copenhagen. There are two datasets called GENSUS and INT99 [38]. These datasets contain very fine 7000 normal ECGs. GENSUS and INT99 contain 10 seconds ECGs signal recorded by 8 lead channels.

[1] https://www.physionet.org/content/ptb-xl/1.0.2/

3.1.1 Data Preparation

Before training the model with real data, we will perform some pre-processing steps. Although doctors use 12 lead ECGs signals for analysis, in reality that information is captured by 8 leads and 4 of the leads are calculated using mathematical equations as presented from eq. 3.1 to eq. 3.4. Therefore, to train the model we are going to use only 8 lead signals. Based on the information from doctors, these 8 leads namely ['I', 'II', 'V1', 'V2', 'V3', 'V4', 'V5', 'V6'] are real lead used for carpeting data. Therefore, we will select only 8 lead channels for input. After this, we will normalize the dataset by dividing the ECG signal by the maximum value of the ECG signal. The open-source (PTB-XL) dataset and private dataset (INT99 and GENSUS) will be normalized by dividing 35 and 6000 respectively. This will bring the data in the range of -1 to 1.

$$III = II - I \tag{3.1}$$

$$aVR = -0.5 \times I + II \tag{3.2}$$

$$aVl = I - 0.5 \times II \tag{3.3}$$

$$aVR = II - 0.5 \times I \tag{3.4}$$

3.2 Neural Network Selection

The problem we are going to solve is related to multidimensional time-series data. Generation of time series data requires different techniques than tabular dataset because of the dependencies of future data points with the past data points. To deal with these types of problems, normally sequential models (Recurrent Neural Network (RNN)[2], Long Short-Term Memory (LSTM)[3], Gated Recurrent Unit (GRU)[4]) are widely used in practice. However, because of the sequential process (i.e., parallel computation is not possible), using these types of models are costly (in terms of time and hardware cost) for heavy-generation models. As an alternative, researchers use the one-dimensional CNN model with larger kernel size for these types of problems. We can get the benefit of parallel computation using CNN (i.e., CNN are not sequential models). Similarly, to capture the dependencies of present point with the past and future data points, we are going to use attention mechanism[5] and temporal convolution network (TCN)[6]. Therefore, to solve the defined problem, we are going to use mainly 1D CNN, TCN, and attention techniques as neural network models.

3.3 Proposed GANs Architecture

Our first method is based on generative adversarial neural networks (GANs). Instead of starting from scratch, we take a inspiration from [93]'s model (Pulse2Pulse) architecture as a starting point which was authored by co-supervisor (**Vajira Thambawita**) of this thesis work. The trained model can generate 8 lead ECGs signals with a length of 10 seconds. The model did not consider conditional information for generating ECGs. Similarly, after the careful observation of the fake ECGs generated by the Pulse2Pulse model ([93]), doctors realize that

[2] https://apps.dtic.mil/dtic/tr/fulltext/u2/a164453.pdf

[3] http://www.bioinf.jku.at/publications/older/2604.pdf

[4] https://arxiv.org/pdf/1409.1259.pdf

[5] https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[6] https://arxiv.org/pdf/1803.01271.pdf

model is not sufficiently able to generate P-waves. There might be several reasons behind this, however, one of the possible reasons could be not using any techniques to learn past and future dependencies. Therefore, we believe this could be improved by adding a Temporal Convolution Network (TCN) which provides the ability to learn the dependencies of current points with long time past and future time points. To enhance the capability of the generator to generate the required distribution of data, we will also use the normalization technique (Group Normalization). To enforce the model to generate synthetic ECGs of desired features which is passed as a condition, we have designed the model in a way which takes ground truth (conditional) information and converts it into feature space and passes that feature space into each block. The proposed model architecture of the generator is presented in the following Figure 3.1

3.3.1 Generator

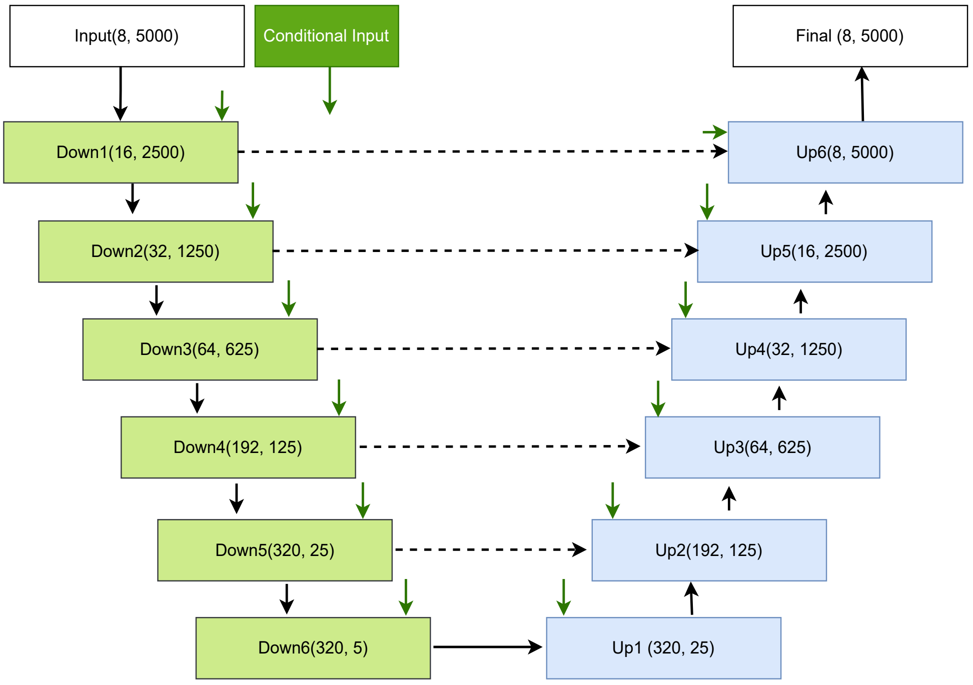


Figure 3.1: Proposed generator model.

As you can see in Figure 3.1, the proposed generator has a U-shaped design inspired by U-NET model architecture [77]. The generator takes uniformly distributed noise between -1 and 1 in the shape of 8 × 5000 and ground truth as inputs. The noise and ground truth are passed through the six layers of encoder blocks (responsible for reducing features) which converts the noise into latent space. After that, the model passes through another six blocks which is also called decoder block (responsible for enlarging features, i.e., the reverse process of encoder block). Each decoder block receives inputs from the previous block and the corresponding level of the encoder block except the first decoder block (UP1). In addition, each decoder block receives ground truth as well. In Figure 3.1, as you can see, the conditional information is passed into each block of architecture represented by the green arrow. Similarly, in the figure, we have named the encoder and decoder blocks as down and up blocks respectively. The details of the layer used in down and up blocks are presented in the following figure.

As demonstrated in Figure 3.2, each down (Encoder) block consists of another three sub-blocks. The first sub-block is called CNN which consists of a single 1D-CNN, group normalization and activation layer. This is the only sub-block in each down block where we

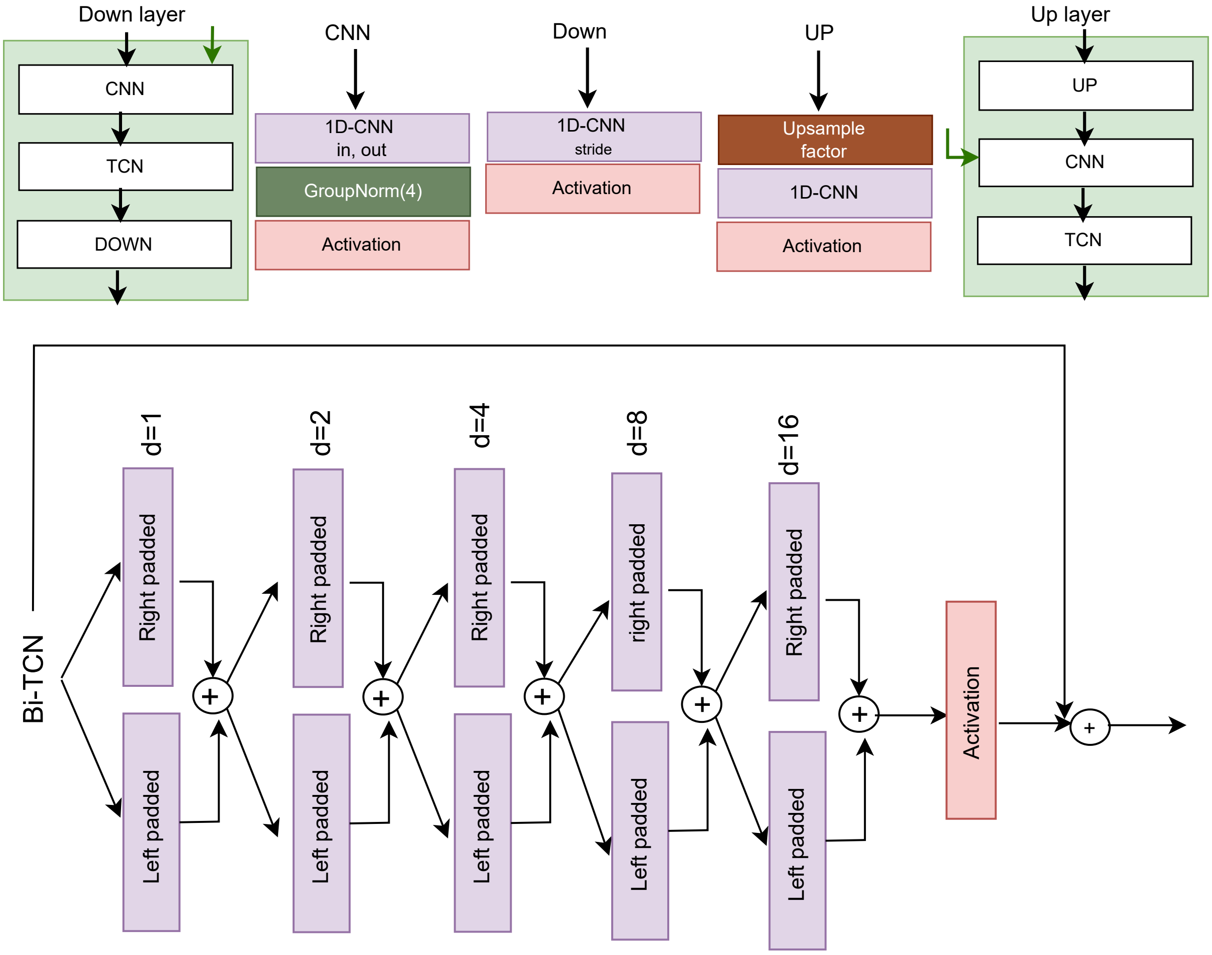


Figure 3.2: Details of layers used each block of proposed GAN model.

change the channel dimensions, i.e., in the rest of two subblocks (TCN and DOWN), input and out dimension remain the same. The second sub-block is called TCN, which is one of the core parts of our proposed model architecture. Temporal convolution network is similar to normal convolution, but it has an additional parameter called dilation which changes kernel learning pattern. Figure 3.3 demonstrates the concept of TCN where kernel size is set to 3 with three dilation rates (1,2,4). The dilation (d=1) is a normal convolution network and convolutional network with dilation of 2, 4, etc. is the concept of TCN. The dilation rate changes in an order of 2^n. This gives the model ability to generate a single data point by looking at the relation with long-term past history. This is the general concept of TCN.

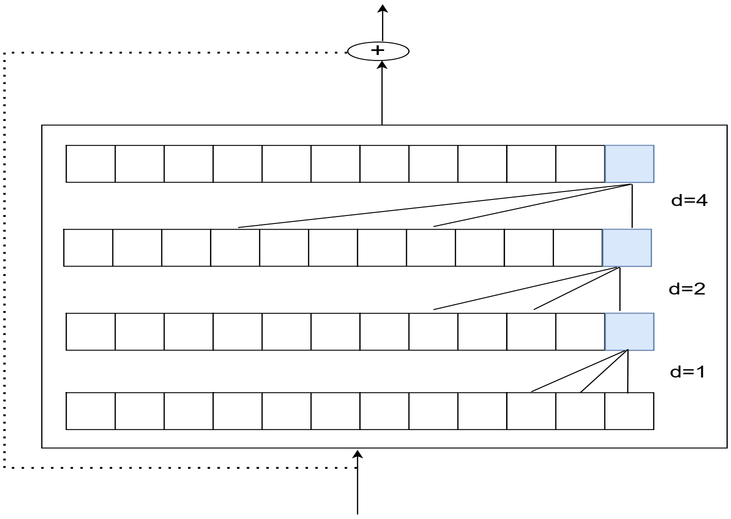


Figure 3.3: Temporal Convolution Network(TCN) architecture with residual connection which is indicated by dotted line.

However, after careful observation, we have realized that the models might have problems in generating starting few data points. This could be because of padding to only the left side in TCN. It might not be a problem in the classifier but in the generative model it is the bottleneck. To overcome this issue, we will use the concept of bidirectional TCN (Bi-TCN) as presented in the bottom figure of Figure 3.2. The core concept of proposed Bi-TCN is that at each dilation rate (d), data passes through two distinct CNN with defined rate of dilation. To generate the data of the same size, the model needs to add padding. Therefore, we are adding zero padding to only the left side (as normal TCN) and another one with padding to only the right side. Then we aggregate the output of two TCN layers and pass it to forward layers. Then, the output of final passes through activation layers. To keep track of information that has been learned in previous layers, we use the residual connection at the end.

The final sub-block of the encoder block is called the down block which is responsible for reducing the time dimension features. As you can see in the top layer of the middle figure in Figure 3.2, the down block has only one 1D-CNN layer with varied stride followed by activation layers. The value of stride decides the output of the layer, for example, stride=2 reduces the feature by half.

Similarly, Up block (Decoder) has similar architecture to down block (Encode). In the down block, the feature is reduced at the last sub-block but in the up block, first it passes through the feature enlargement layer and then passes through CNN and TCN layers. CNN and TCN blocks are the same as in down block. The up-block sub-block has three layers, an upsampling layer (which is interpolation with the same method), 1D-CNN layers and activation layer. The value of the upsampling factor will be the same as the value of stride that is used in the corresponding level of the encoder (down) block. For example, in the last down block, features are reduced by a factor of 5 (in feature=25, out feature =5), therefore, the first up block used the upsampling factor as 5.

The final block in the Figure 3.1 contains the final 1D-CNN layer to reduce the effect of noise that has been added from the corresponding down block (i.e., down1). The layer takes input in the shape of (8 x 5000) and output in the same which is the required output shape. The output of the final layer then passed through the tanh activation.

3.3.2 Generator Variants

The model architecture presented in Figure 3.1 is our proposed base model. We have named our base model Advance Pulse2Pulse (*AdvP2P*). We have made some assumptions in our based model and added different techniques on top of the based model to further enhance the generation of data. We briefly explain the core concept of each variant of generator in the section below with their proposed names.

3.3.2.1 AdvP2P-SD

We claimed that the use of bi-directional temporal convolution network (Bi-TCN) can generate better quality of ECG data. To verify this hypothesis, we have built a single dimension (normal) TCN model which we have named Advance Pulse2Pulse - Single Dimension (AdvP2P-SD).

3.3.2.2 AdvP2P

We have called our base model Advance Pulse2Pulse (*AdvP2P*). This is the exact model that we have presented in Figure 3.1. Our proposed base model uses bi-directional TCN.

3.3.2.3 AdvP2P-PosEmb

Instead of directly passing the random noise in the U-NET base generator, if we change the random noise based on its position (time-dimension) by using positional embedding [97] (also called sinusoidal position embedding) which will convert the random noise into smooth form. We believe this method will help to improve the quality of data. In our proposed base model, we add this layer just after the input before it passes to convolution layers. We have named this version of model as Advance Pulse2Pulse - Positional embedding (*AdvP2P-PosEmb*).

3.3.2.4 AdvP2P-AutoEmb

Positional embedding is the combination of sine and cosine waves. Adding those waves into random noise helps to improve the generation process but this may not be the optimal solution as ECG has different styles of waves than sine and cosine. Therefore, instead of using fixed positional embedding, we let the model learn the required embedding by itself before it passes to the convolution layer. The generation process may become easier. This is used the same as positional embedding i.e., just after the input layer. We have named this version of model as Advance Pulse2Pulse-Auto Embedding (*AdvP2P-AutoEmb*).

3.3.2.5 AdvP2P-NtF-AutoEmb

Another trick that we have added on top of the previous model (*AdvP2P-AutoEmb*) is the idea of converting random noise into features and then passing through the auto (learned) embedding. We have added another layer called NtF (Noise to Feature converter) before Auto Embedding layers. The layer architecture of NtF is presented in Figure 3.4.

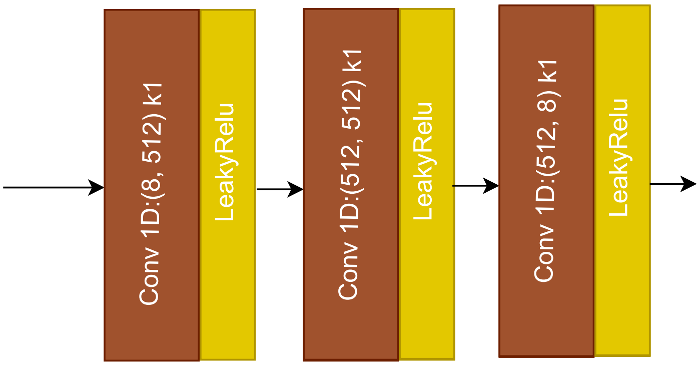


Figure 3.4: Noise to feature converter block.

3.3.3 Discriminator

In the domain of GAN, researchers have more focus on designing generator models than the discriminator. The discriminator is a simple classifier model which classifies whether an input is fake or real. In the thesis work, we have tested different versions of discriminator specially PatchGAN [43] based discriminator, but it does not work as expected. We will explain more about it in the discussion section. Finally, we adopted the discriminator of Pulse2Pulse model and added the conditional factor in the model. We first convert the target information (also called ground truth interchangeably) into a feature space of 5000 by simply repeating the same value. Then, we concatenated target information with real and fake samples before it passed to convolution layers. The architecture of the discriminator is presented in the following Figure 3.5.

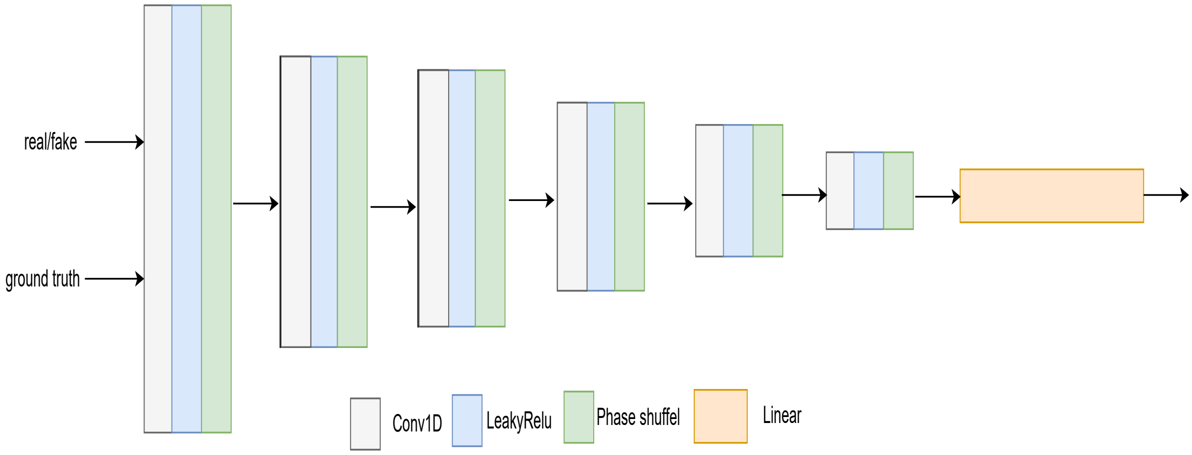


Figure 3.5: Discriminator architecture.

3.3.4 Loss

We have already written the section about the BCE loss function (in section 2.3.4.2), its problem (in section 2.3.4.3) and the optimized loss function used in state-of-the-art models (in section 2.3.4.4). In this thesis work, we will use the Wasserstein loss which is the commonly used loss function in recent state-of-the-art models. We recommend reading the background section to know more about the Wasserstein loss. To make the discriminator more generalized, we will also use the technique of gradient penalty while training the discriminator.

3.4 Proposed Denoising Diffusion Model Architecture

While going through the detail literature review, we figure out the many pitfall of training GANs reported by researchers [79]. For example, time consuming, requires advance hardware (face memory issues with smaller GPU), strong model parameter dependencies on the final result, small modification on model leads to something random/unexpected output, something may not be able to get similar result in second training process etc. Those hurdles encourage us to research the alternatives of GANs. At the same time, in 2021, the researchers from UC Berkeley published a research paper based on denoising diffusion model and authors argued that their model beats the state-of-the-art GANs model in generating images. Authors claim that diffusion models are state-of-the-art generative models in image generation. Similarly, authors reported that training diffusion models are much easier than training GANs models.

To identify whether denoising diffusion model can also generate high quality of realistic ECGs signal and to mitigate the hurdles of training GANs, we work on our second method based on diffusion model. Our model architecture is inspired by [39]. The authors have proposed diffusion model for image generation. We took the reference from this model and proposed modified version of architecture for generating multidimensional ECGs signals.

3.4.1 Forward Process

The forward process of diffusion model is static and there is no need to calculate any gradients. We need to define some of the model parameters like time steps (number of times we want to add noise in the real data), noise range (minimum amount of noise we want to add to maximum amount of noise). Once we have defined these parameters, in the forward process. we add fixed amount of noise in each time steps and make data noisy. In the Figure 3.6, the top to bottom steps refers the forward process where X_0 represents the real data and X_t represents how the data looks like at that time steps. We can see that the last time steps (i.e., t=240, it could be any number, best practice is to use around 500 to 1000), fine plot of ECGs (at t=0) converge into complete noise.

3.4.2 Backward Process

The backward process is reverse operation of forward process. The main concept of backward process is that if it is possible to make data noisy after adding predefined amount of noise at every time steps, it should be possible to estimate the amount of noise added in each step and removed that noise from data which will intimately converge into real data. In the figure 3.6, bottom to top process is backward operation where the model predicts the X_{t-1} based on X_t.

In the training of diffusion models, the size of data will remain the same. Therefore, Unet model architecture is used in the backward process of diffusion model. The Unet architecture used in our work is presented in the Figure 3.7.

The model takes real ECGs with the shape of $8x5000$ as input and goes the initial 1D convolution layer (with kernel_size=25 and padding=same) which convert the input into $64X5000$. Then it passes into the four layers of downsampling steps. Each step has four different layers. The first two layers are residual blocks, the third layer is residual connected attention layer and last layer is downsampling steps. How the number of channels and data size is changed over each block and layers is presented in the figure.

The architecture of the first two residual blocks in each layer is identical which is presented in Figure 3.8. This block accepts two inputs: data and time steps. As you can see in the figure, residual architecture has two blocks of CNN where the first block accepts data and time embedding information. The second block receives the output from the first block. To keep track of what it has learned before, it used residual connection. This type of technique helps to

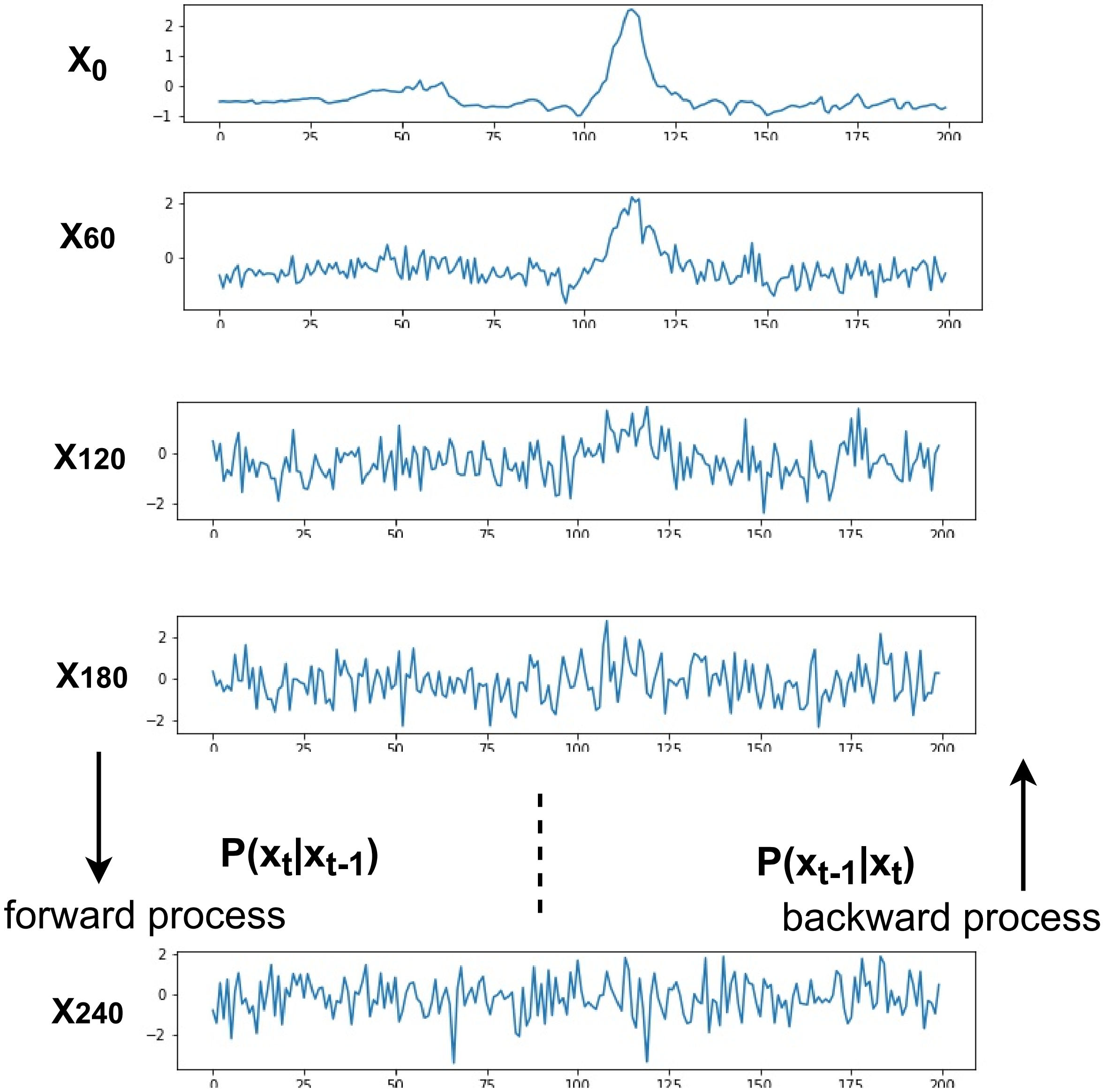


Figure 3.6: Forward and backward process of diffusion model training.

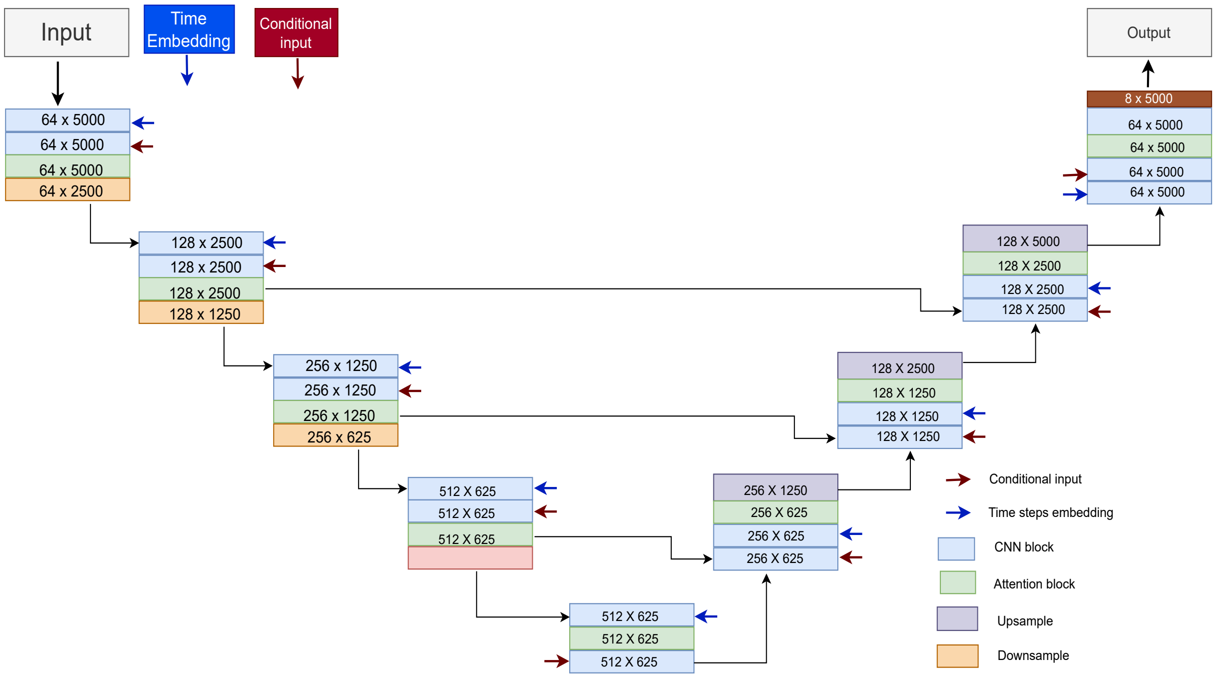


Figure 3.7: Unet architecture used in diffusion model.

improve the model in better learning. The right side of block represents the CNN block used in residual block (left). The solid line represents the noise data and dotted line represents the time embedding information. In the CNN block, input passes through 1D CNN layer, then group normalization and if the time embedding information is available, it combines the embedding information with input in an interpolation way. This step is especially important because based on this time embedding information, the model predicts the next time step data. Therefore, we can consider it as a major step in diffusion model. Finally, LeakyReLU is used as the activation function in the last layer.

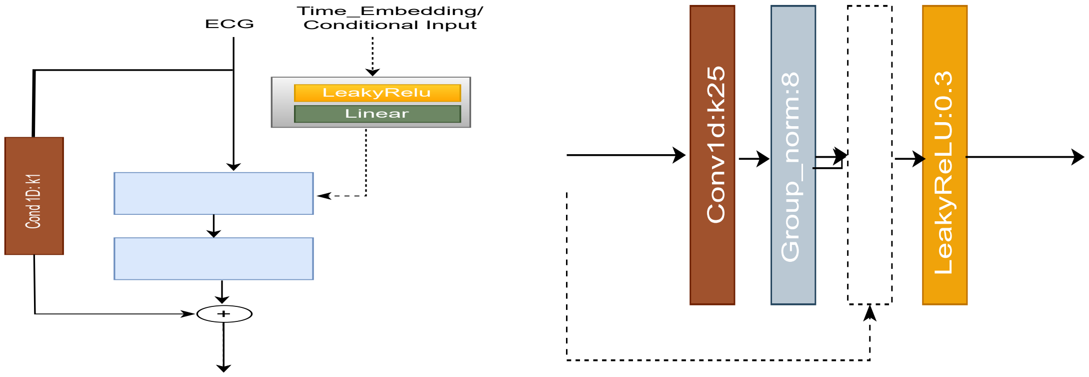


Figure 3.8: (Left) Layer architecture of residual block and (Right) layer architecture of CNN block used in residual block.

After the first two block of residual, the third block is residual attention which is presented in Figure 3.9. This block holds the residual connection and attention block. First, input passes through layer normalization, then it goes to attention layer, and finally the output is added with starting input as residual connection. Layer normalization helps to normalize the data better than batch normalization as it can perform better in lower batch size as well.

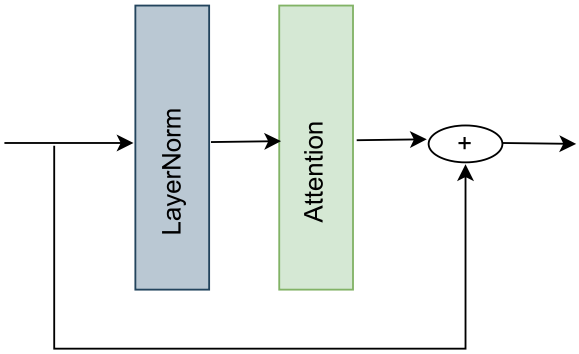


Figure 3.9: Residual attention block.

We have used two types of attention. In the down-sampling blocks and up-sampling block, we have used attention block while in the middle block, we have used linear attention blocks. The computation difference between two blocks is presented in Figures 3.10 and 3.11 The concept of linear attention helps to boost the model performance and at the same time it is computationally efficient [54].

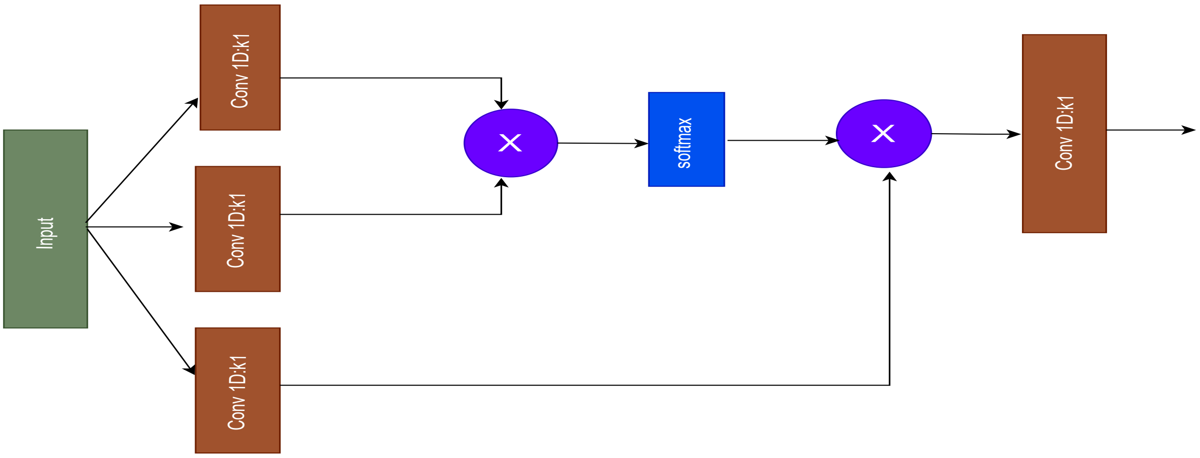


Figure 3.10: Attention layers architecture.

3.4.3 Algorithms

We have explained the forward and backward process theoretically in the earlier sections. This section explains pseudo steps of the training and sampling process. To make you easy to understand the different mathematical symbols used in training 1 and sampling 2 algorithm, all the mathematical symbols are presented below.

- T = `Total number of time steps noise added (500)`
- β = `list of noise added at each time step, low=0, high=0.99`
- $\alpha = 1 - \beta$

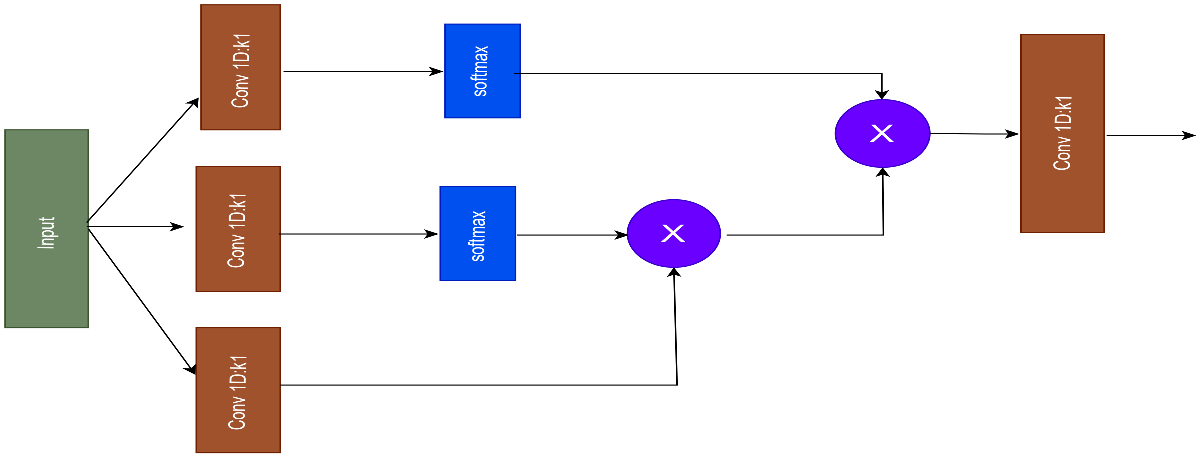


Figure 3.11: Linear attention layers architecture.

- $\bar{\alpha} = \prod_1^T \alpha$ i.e. cumprod(α)
- $x_0 = input$
- $noise(\epsilon) \sim \mathcal{N}(0, I)$
- $\sigma_t = \log(\beta * \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t})$
- $\epsilon_\theta = model$

The training steps presented in Algorithm 1 shows that at each iteration, the input x_0 is converted into noise by adding noise t times where t one of the times points randomly selected from the range of max time steps $(1, .., T)$. The noised input $(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t)$ is then go to the model ϵ_θ which predict the amount of noise added at that time step t. Then the loss on predicting noise will be calculated and update the model parameters based on that loss. This process continues until the model is converged.

Algorithm 1 Training [39]

```
repeat
$x_0 \sim q(x_0)$
$t \sim Uniform(1, ..., T)$
$\epsilon \sim \mathcal{N}(0, I)$
Take gradient descent step on $\nabla_\theta ||\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t)||^2$
until converged
```

The sampling (generating new sample) process is bit different than training steps which is presented in Algorithm 2. It starts with random noise x_T and does the iterative process from reverse order (i.e., from last time point to the first time point). At each time point t, the model predicts the amount of noise added at that point t, then the model reduces the predicted noise from current time of x_t. The process continues until it reaches the time unit of 1.

3.4.4 Loss

Calculation of loss is one of the important steps in training the model. Loss function guides the model to better learn the relationship between input and output. In the case of diffusion, the model will re-generate the real sample back. Therefore, we use l1_loss during training model.

Algorithm 2 Sampling [39]

```
$x_T \sim \mathcal{N}(0, I)$
for t= T, ...,1 do
    $z \sim \mathcal{N}(0, I)$ if t > 1, else z=0
    $x_{t-1} = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t) + \sigma_t z$
end for
return $x_0$
```

The L1 loss is calculated by differencing absolute difference between the real and predicted data. L1 loss function is also called mean absolute error (MAE).

3.5 Condition Encoding

Condition encoding is another major step to address the problem of conditional data generation. How we pass the condition information to the model has significant impact on the out of the model. Traditionally, one-hot encoding or embedding layer is used for condition values and concatenated with input. Several research shows that these types of methods work. However, it is not able to pass optimal information as the information is only added as layer. On the other hand, in new state-of-the-art work (e.g.: StyleGAN), interpolation techniques have been used which are able to get optimal result in transferring styles. Therefore, we have used similar techniques for conditional encoding.

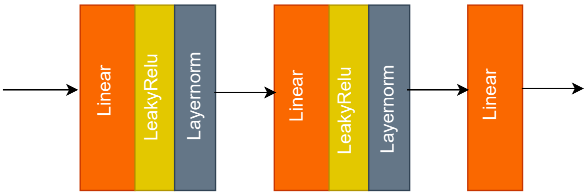


Figure 3.12: Encoding conditional Information.

The architecture of conditional encoding is presented in Figure 3.12. First, conditional information (also called target information) is passed to Embedding layer then it passes through linear, leakyrelu activation and linear layer correspondingly. After encoding this information, it is combined with time embedding information. Time embedding is also done in an analogous way to condition encoding. The only difference is that in the case of Embedding layer, Positional Embedding is used in Time embedding. The combined information is passed to the data as shown in Figure 3.13.

3.6 Analysis of Outcomes

Verifying the quality of data generated by our proposed methods is a crucial step in the generation of synthetic data. Training the model successfully will indicate that our proposed models will be able to generate similar kinds of data exits in the training set. However, it is very important to verify that the proposed models are able to capture the different kinds of features that exist in the dataset. For example, in our case, the generated ECG data must capture P-wave, QRS complex, T-interval, etc. For this purpose, we are going to follow the different kinds of steps which are discussed below.

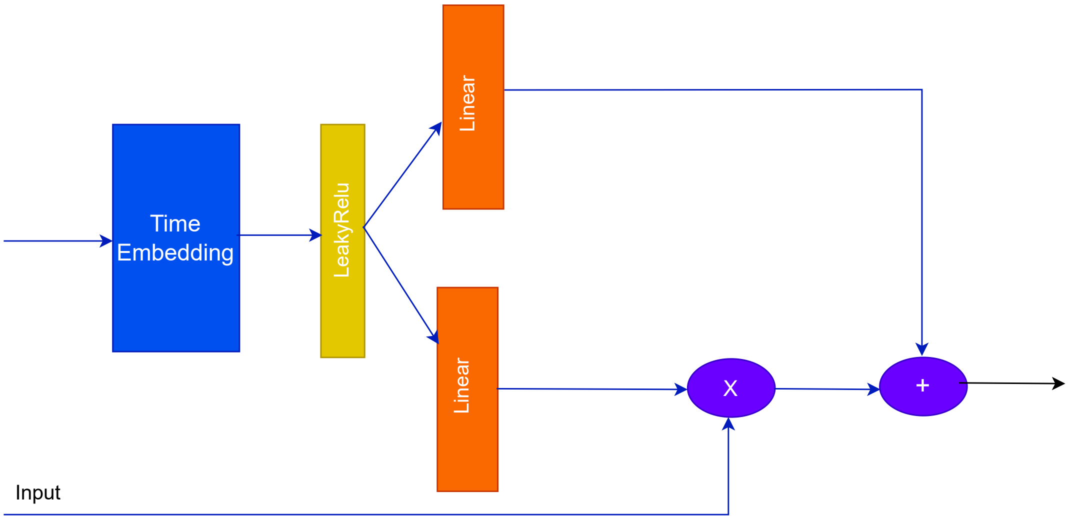


Figure 3.13: Condition information passing techniques. The mathematical terms × and + refers to element wise multiplication and additions respectively.

3.6.1 Loss Analysis

Analyzing the loss of different models is a simple and effective and widely used method for analyzing the performance of models. In our case, loss refers to discriminator loss (more specially it will be Wasserstein loss) in the case of GAN models. The Wasserstein loss will tell us the actual difference between real samples and fake samples over the period of training. Lowering the value of Wasserstein loss refers to better quality of fake samples and vice versa. On the other hand, in the case of diffusion model, the model will predict the amount of noise added on selected time steps during the training phase. It is important to understand that at each iteration, the model predicts the noise of only one randomly selected time point, i.e., it will not predict the noise of every time steps T to 0. Therefore, it will not give meaningful insides of generated data.

3.6.2 Visual Inspection

As our first evaluation steps, we will generate graphs of real ECGs and fake ECGs graphs and verify ourselves whether our proposed model is able to generate good enough data or not. As we are not experts of ECGs, our main focus will be on checking visual differences in the ECG graphs. For example, heart rate pattern, lower and higher amplitudes etc. For the visualization of ECG, we will use the tools (*ecg_plot*) used by doctors. *ecg_plot* is a python based open-source package.

3.6.3 Similarity Score

Similarity score is one of the key methods for checking the quality of generated fake samples. There are various kinds of similarity function, however, in comparing the result of generative models, Inception Score (IS) and Frechet Inception Distance (FID) are two mostly used methods. As the name suggests, these two methods are based on a deep learning model called inception classifier for classifying images [89]. Instead of checking the similarity of samples in original size, a pre-trained inception model extracts the class labels or lower dimensional features from both real and fake samples, and based on these data two different similarity score will be calculated. The higher the value of IS is considered to be better which will depend on two thing 1) lower entropy score - if class label belongs to single class 2) evenly distributed class labels - data belongs to different classes in equal distribution. Similarly, lower the value of FID is considered better which compare the distribution of real and fake sample class labels. FID is more reliable than IS because IS does not take account of real data.

To calculate IS and FID, we need to prepare our ECGs signal into three dimensions (3, 299, 299). To convert our ECGs signal into required form, fist we flatten the ECG signal (8 x 5000) into 40,000 points. Then, we will reshape data into (200 × 200). We will copy the same data three times and make it 3 dimensional (3, 200, 200). Finally, to convert the data into the required width and height of 224, we will use interpolation methods. Both the real and fake samples pass through our created data pipeline (flatten, reshape, expand, interpolation) and then goes to inception model as input. From the inception model, we will extract 2024 points, and calculate IS and FID using built in function available in *torchmetrics*.

3.6.4 ML Classifier Metrics

Similarity scores give us information about how much real and fake samples are close to each other. However, based on similarity score we cannot say whether fake samples look real or not. On the other hand, it is impossible for humans to classify an enormous number of samples into real or fake. Therefore, we will use a machine learning approach to classify the data into real and fake samples. This is a more convenient process because machine learning models are good at identifying patterns from dataset. The main expectation of this method is that if both datasets have similar patterns, then it will be difficult for the model to classify between real and fake which results in lower accuracy and vice versa.

To calculate below discussed metrics, we will use 10,000 fake samples generated from each model and 10,000 real samples in case of ptb-xl dataset and 7073 real samples from private dataset. We will use 50% of the data for training purposes and 50% for testing purposes. To make sure that both data exist in training and testing in equal proportion, we will use stratify split. Regarding the model, we will use transfer learning method on pre-trained resnet50 model to classify real and fake. We will prepare our data in a similar way as we used similarity scores. The only difference is that resnet50 uses 3 X 224 x 224 size of data instead of 3 × 299 × 299.

More specifically, from the classifier model, we will check three different metrics, namely, Accuracy (AC), Precision Score (PS) and Recall Score (RS). The abbreviation used to explain these metrics in below formula are as follows:

- TP = True positive (correctly classified real)
- FP = False positive (misclassified fake as real)
- TN = True Negative (correctly classified fake)
- FN = False Negative (misclassified real as fake)

$$Ac = (TP + TN)/(TP + FP + TN + FN) \tag{3.5}$$

$$PS = TP/(TP + FP) \tag{3.6}$$

$$RS = TP/(TP + FN) \tag{3.7}$$

Accuracy (AC): Accuracy identify how correctly real, and fake are classified. Higher value refers to correctly distinguishing real and fake samples and vice versa. Accuracy is calculate using following formula presentd in eq. 3.5.

Precision Score (PS): Knowing the accuracy gives an overall overview of how many samples are correctly classified, however, it does not reveal which class of data is mostly misclassified. In our case, it is very important to know who many fake samples are also classified as real (i.e., falsely classified as positive) which is exactly captured by precision score. PS calculates the accuracy of positive prediction using the formula presented in eq. 3.6.

Recall Score (RS): Recall score, on the other hand identified how many real samples are classified as fake (i.e., false negative). RS identifies the proportion of actual positive classified correctly using the the formula presented in eq. 3.7. Higher the value of recall indicates that most of the real are classified as real and vice-versa. In our case, it is considered to be an ideal condition when the value of RS drops around 0.5.

3.6.5 Power Spectrum Analysis (PSA)

Similarity score gives information about how close real and fake samples are, but it does not tell us whether that level of close is good enough or not. In fact, we cannot get any idea about whether different kinds of waves are correctly generated or not. On the other hand, ml classifier metrics bit more information especially if the accuracy, precision and recall score drop around 50 which is ideal condition. But getting such a result is extremely hard. If the model does an excellent job at classifying, then we cannot say where the problem is. Maybe some of the waves are correctly generated or may have problems in all waves. To answer all these questions, power spectrum analysis is a useful tool. PSA converts the time domain information into frequency and its corresponding strength. Comparing this two information from real and fake samples will help to identify whether the frequencies that exist in real data are also present in fake samples with their corresponding strength. If there are differences in between the strength of real and fake samples at particular frequency, it will tell us that particular wave is smaller or larger. We will make this comparison at channel level based on mean value.

3.6.6 Latent Space Visualization

Visualizing the data at higher dimensional and doing analysis and making decision on top of it is difficult. For example, based on only visual inspection (high dimensional) method, it is difficult for anyone to say how much they differ from each other. To mitigate this gap, latent space visualization is another method for comparing the real and fake samples in latent space. Here the latent space refers to latent features or also called low dimensional features.

To complete this analysis, we will first build and train an autoencoder model. After the model is trained, we will use a trained model to extract the latent space only. The design autoencoder model will convert the ECG signal of 8 x 5000 points to latent space of 500 points. From 500 points, we will use the first 250points as x-axis and last 250 points as y-axis so that we do not have to compress further. As we are working on two datasets, instead of training the autoencoder model in two datasets, we will train the model on only one dataset (PTB-XL) and use it for analysis to both datasets.

3.6.7 Verify by Experts

We know some of the features of ECG, however, we are not experts in the ECG field. It is very important to know that generated fake ECG are non-differentiable. At the same time, it should capture the properties of ECG. As we have a group of expert doctors connected with our project. We will verify our result with expert doctors. For this purpose, we will generate an ECG graph of 100 samples from each model and provide it to doctors. Doctors will look at the ECG graphs and try to observe the overall quality of fake samples and which model samples look more promising. If we consider our fake samples to be used in real application, this step is especially important. Doctors feedback can be used as a suggestion to improve the model in future as well.

3.6.8 MUSE Feature Analysis/ ECG Parameters

All above methods will analyze the ECG at signal level which will give lots of information for the comparison of real and fake signals. However, in real life applications, doctors extract the different biomarkers from ECGs signal and use that information for analysis. Here the biomarker refers to features related to ECG like R peaks, T-peaks, QT-interval, etc. Comparing the different features of ECG using statistical tools is more convenient than comparing at visual way. For this purpose, we will use MUSE tool to extract different biomarkers from real and fake signals. MUSE is one of the widely used tools by doctors at hospitals. This is not an open source. Therefore, we will request our doctor to generate statistics for us. After we get the generated statistic, we will analyze the differences between real and fake ECG features.

Initially, we planned to use MUSE tool as explained above, however, the real system is very busy and our data were on queues. Therefore, as an alternative, we will use AI based tool for extracting ECG parameters which is trained on the same private dataset [38]. The AI-based model is compatible with MUSE tool. The real MUSE system can extract large features whereas AI-based tool can extract 7 different features.

3.7 Summary

In this section, we present our proposed model architecture based on GANs and diffusion. To make it easy to understand the proposed model architecture, we have presented an overall architecture and each component of architecture graphically. We made different hypothetical assumption and based on that proposed different version of GANs model. We used extensive research to figured out an efficient way to pass the conditional information. Based on our research, we added the conditional parameters in the proposed generative models. In the case of diffusion model, we did research on the possibilities of using image based model to time series based model. Similarly, adding the conditional input in more effective way to the existing model architecture is another effort we made in proposing models architecture. The evaluation of generated synthetic data compared to real data is one of the challenging tasks in the validation of synthetic data. Therefore, we have presented different approaches for validation so that we can draw better conclusions. All the models proposed in this section will be used for experiments and generating data in the next section. The data generated by proposed models (in Section 4) will be evaluated based on define evaluation methods described in this section.

Chapter 4

Experiments and Results

In this section, we are going to explain the experimental set and results from different experiments and models. As presented in the methodology section, we are going to use different techniques for result evaluation so that we can make more reliable decisions. Therefore, the results are presented in different sections.

4.1 Experimental Setup

This section gives you the overview of how the experiments have been conducted in this thesis work. We have two different generative model approaches. The experimental setup of each approach differs from each other. There are some similarities as well. For example, all the experiments were conducted on two different servers. For the open-dataset (PTB-XL), we have used the eX3 server provided by Simula Research Laboratory regardless of model approach. There are different types of clusters in the server, however, all of our experiments are conducted on a server which holds Tesla V100 GPU of 32GB. Similarly, for the private dataset provided by University of Copenhagen (UiC) (INT99 and GENSUS), we have used the servers provided by UiC via VPN service as we are not allowed to work on data out of server. The server holds GeForce RTX 3090 Ti of 24 GB. Without use of these resources, all the experiments conducted in this thesis work is not possible. Therefore, we would like to thank you both organizations for providing access to the great resources. In the sub-section, we explain our training strategy. For more technical details, we recommend you check the GitHub page of project by following this url: https://github.com/upretiramesh/SyntheticECG.

In following experimental result, you might find some term used interchangeably, for example, open dataset represent PTB-XL dataset, private dataset represent INT99+GENSUS dataset.

4.1.1 GAN

During the training of GAN models, we have set the learning rate (lr) of 0.0001 with beta1=0.5 and beta2=0.9 in both of the optimizer. All the models are trained of 2500 epoch in PTB-XL dataset while 4000 in private dataset. Similarly, for analyzing the performance of models over epochs, we have saved model checkpoints at every 25 epochs. The training strategy is set up in a way that discriminator is trained on every iteration while the generator is only trained on each 2nd iteration (on even iteration only). This helps discriminators to become more and more accurate in distinguishing real and fake samples. In the case of private dataset (INT99 + GENSUS), because of lack of server availability we have only trained AdvP2P-AutoEmb and AdvP2P-NtF-AutoEmb models. We have leaved empty cell for other models.

4.1.2 Diffusion

We have used the same training strategy as used in denoising diffusion models papers. For your easiness, we would like to mention some of the important parameters. We have used time steps of 500 with the beat schedule of cosine methods (alternative is linear). We have used a batch size of 32 and a learning rate of 0.0001. Similarly, *L1_loss* is used as loss function to calculate the during predicting of noise. Diffusion models are trained for 200k iteration and model checkpoints are saved at each 1000 iteration.

4.2 Used Model Abbreviations

We have already explained the different version of our proposed generator in the methodology section with their name abbreviation. In the rest of the below sections, we have used model abbreviation in all graphs to make the things easy and consistent. Therefore, summarizing all the models name with their abbreviation, makes you easy for connecting the used model abbreviation with their associated models.

Model Abbeviation	**Types**	**Model Details**
ECG-DDM	Diffusion	Denosing Diffusion Model
AdvP2P-SD	GAN	Advance Pulse2Pulse with Normal TCN (Single Dimention)
AdvP2P	GAN	Advance Pulse2Pulse
AdvP2P-PosEmb	GAN	Advance Pulse2Pulse with Positional Embedding (sinusoidal)
AdvP2P-AutoEmb	GAN	Advance Pulse2Pulse with self-Learned (Auto) Embedding
AdvP2P-NtF-AutoEmb	GAN	Advance Pulse2Pulse - Noise To Feature Converter- self-Learned (Auto) Embedding

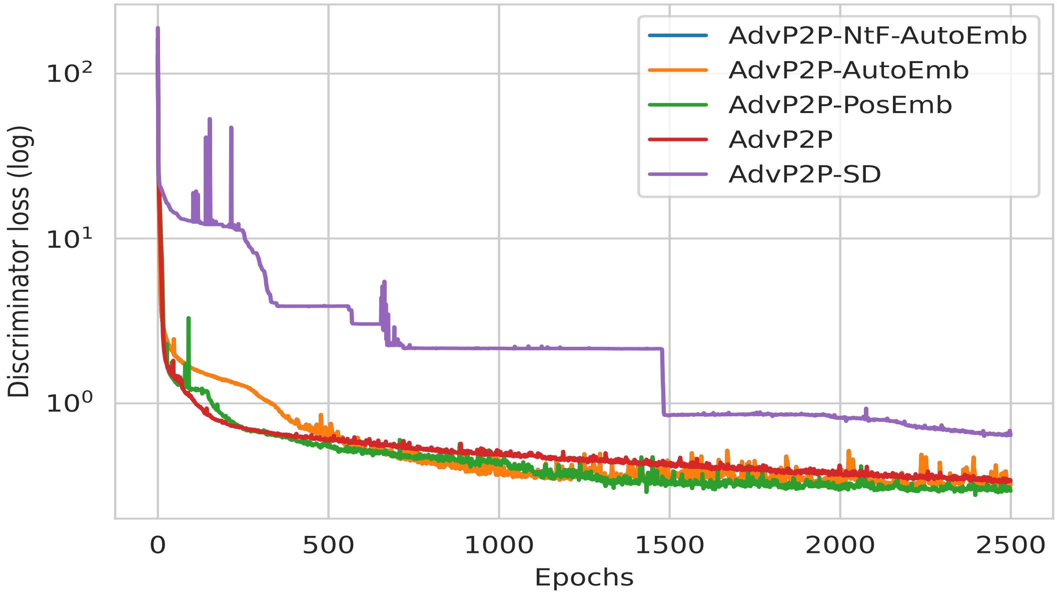
Table 4.1: Model abbreviation.

4.3 Discriminator Loss

After successfully training different variants of generator model under two settings (normal and conditional), we have generated the graph of discriminator loss (wasseratian loss) from the training process. The discriminator loss demonstrated in Figure 4.1 is generated from PTB-XL dataset.

The above discriminator loss reveals two interesting pieces of information about the training process. The first information is that all the models trained on normal mode (without any conditional information), the loss is sharply reduced in first one two epoch and then gradually decrease and reached to the value of 1 around 100 epochs. The losses in between 500 to 2500 epochs show that there is steady decline in loss. On the other hand, on conditional models, the loss drops from peak to level of 1 almost in first epoch, from 2 to 500 epochs, there is gradual decline. And if you compare the losses at 500 epoch and at 2500 epoch, there are very marginal differences which shows that conditional models are good at generating realistic fake samples in smaller value of epochs as well.

Similarly, if we compare the loses of different models in two different training setting, we can clearly see that the claimed that we did during proposing models, i.e., the Bi-TCN based models can generate better quality of fake samples, this argument seems to be right based on the loss. In both conditions, Bi-TCN based models have smaller losses than normal TCN



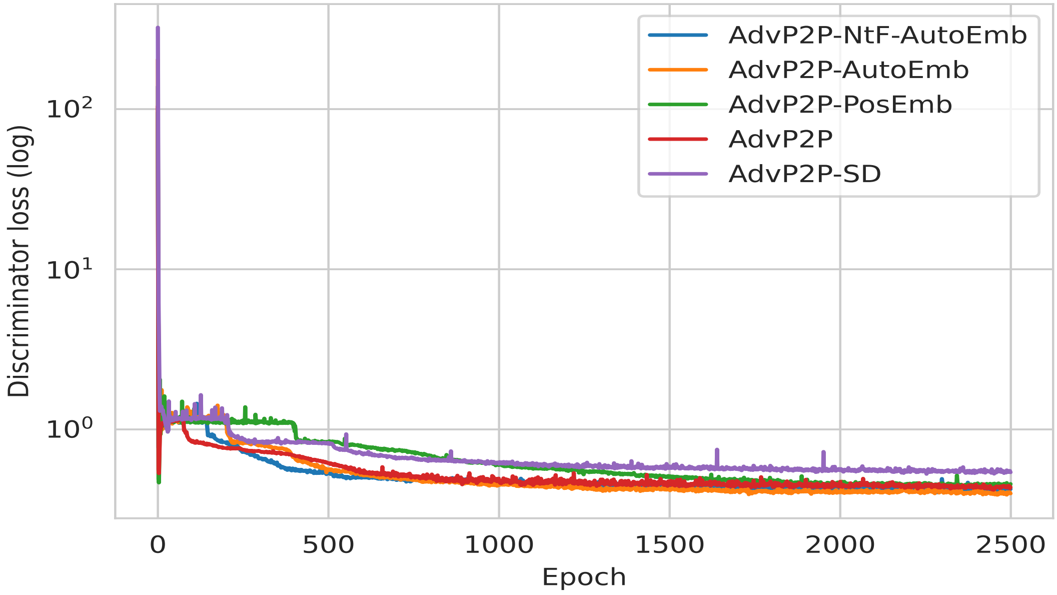


Figure 4.1: Discriminator loss (wasseratian) of (TOP):Normal models, (BOTTOM): Conditional models. Bi-TCN based model (AdvP2P) better than normal TCN (AdvP2P-SD). AdvP2P model is even improved by adding embedding layer (AdvP2P-AutoEmb, etc.).

base model (AdvP2P-SD). Interestingly, the gap the between losses in between Bi-TCN based models and TCN based models are higher in normal training setting, on the other hand, the difference is smaller in conditional training setting. Based on this information, we can say that adding conditional information helps generator to generate required distribution of data.

Last but not least, another assumption that was made, i.e., adding embedding layers (positional/self-learned) to input helps generator to generate better samples seems to be right. If we closely observe the loss on both training settings, embedding based models have lower losses as compared to others. Model *AdvP2P-PosEmb* have lowest loss value on normal training condition while *AdvP2P-AutoEmb* in conditional training mode.

4.4 Visual Inspection

In this section, we are presenting ECG graphs from different settings (real, diffusion, gan and conditional gan) so that we can compare how the real and synthetic samples look in ECG plot view. All of our generative models generate 8 lead ECG, however, in this section, we are presenting 12 leads. We have explained how the rest of the four leads were generated in section **??**. This is the standard way used by doctors in analyzing ECG graphs. The intention of this section is not to analyze the technical details of ECG but to see whether it has captured most of the properties of ECG or not. We can say this section as layman view on fake samples. We have another section below where we will provide expert feedback on fake samples from different models.

4.4.1 Real

The ECG graph presented in Figures 4.2 and 4.3 are generated from real dataset of private and open-source dataset respectively. As we have described in background section, we can see the QRS complex (which is the highest peaks exists in both the graphs), w-wave just after the R peaks and small P-wave before the R peaks. However, P-wave and W-wave do not exist in the chosen sample. This also gives idea that in some of the cases, P-wave and W-wave may not be presented in ECG.

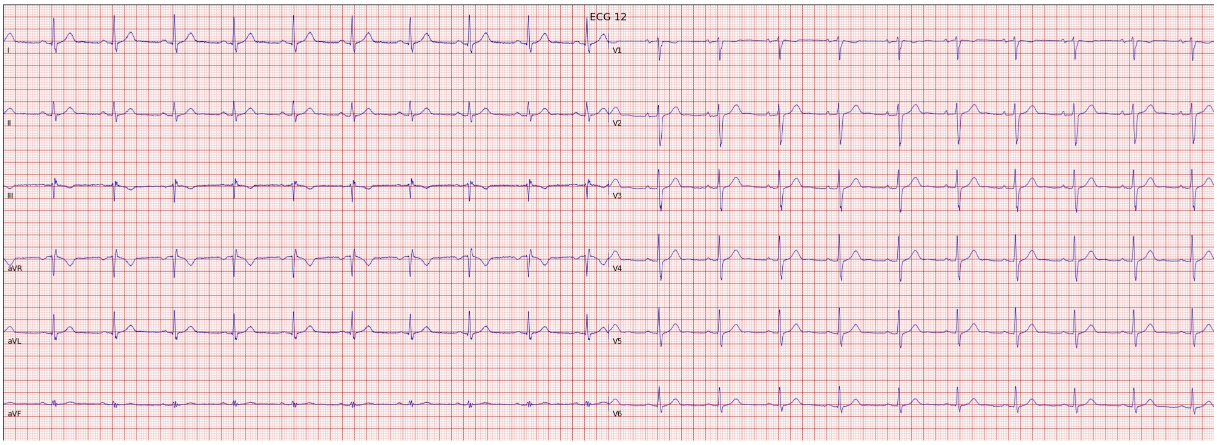


Figure 4.2: Real ECG graph from private dataset(Int99, Gensus)

4.4.2 Diffusion

The graphs presented in Figures 4.4 and 4.5 are generated conditional diffusion models from two different dataset. Both of the graphs have nicely generated QRS complex followed by

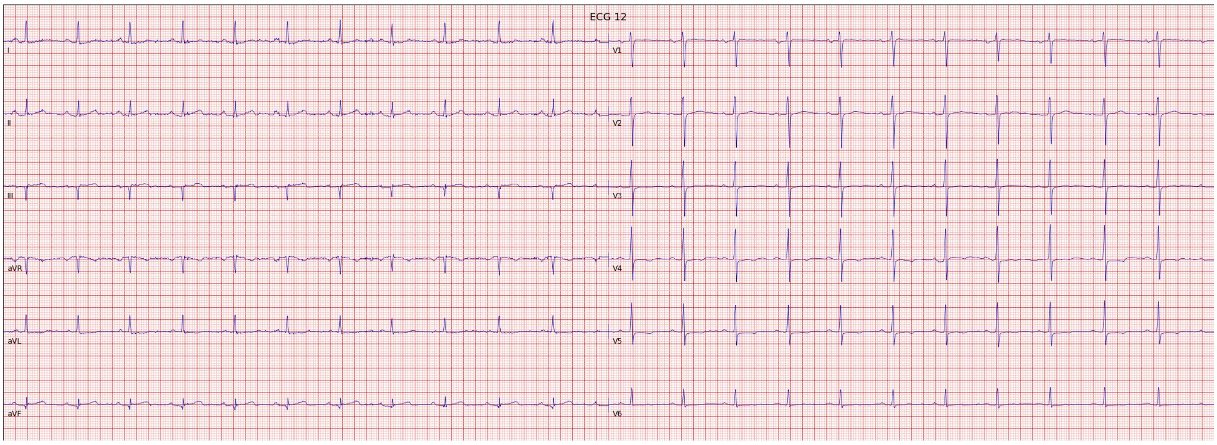


Figure 4.3: Real ECG graph from open source dataset (PTB-XL).

W-wave. In the case of private dataset, small P-wave also exists.

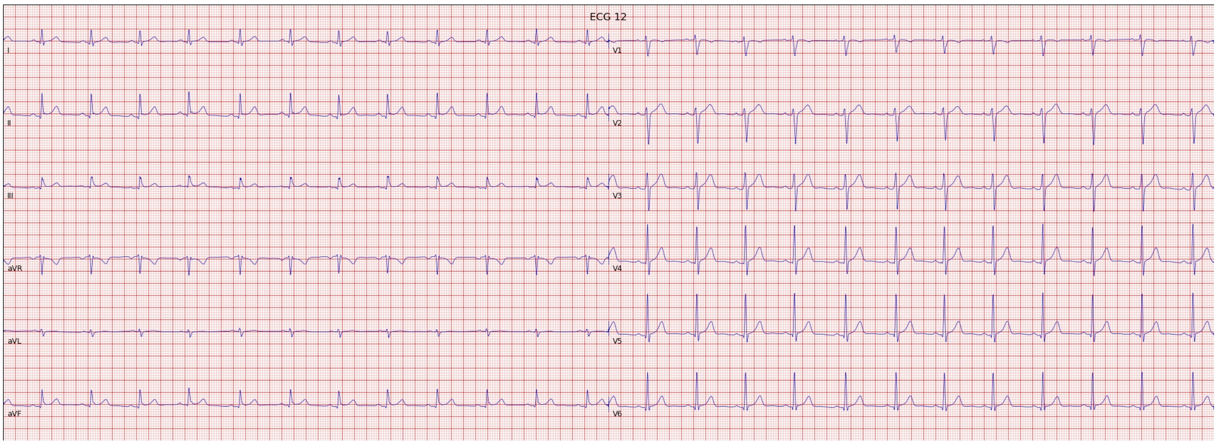


Figure 4.4: Synthetic ECG generated by conditional diffusion model - trained on private dataset.

4.4.3 GAN

Out of five different generator models, we have presented graphs from best model in Figures 4.6 and 4.7 from private and public dataset, respectively. The ECG plot seems very promising. If you closely compare the GAN based models ECG plot with diffusion-based models, it seems GAN based models are good at generating different peaks strongly. However, in diffusion model, different waves seem to be weaker. There might be problems with diffusion or GAN, or it is just because of random plot. We will need to verify this with different statistics and expert review.

4.4.4 Conditional-GAN

As normal GAN models, conditional GAN models are also good at generating realistic fake samples in both of the datasets. The preliminary analysis of these plots shows that models have learned the condition information. However, it has not verified how correctly the model learned the given condition and whether the adding conditional information helps to generate

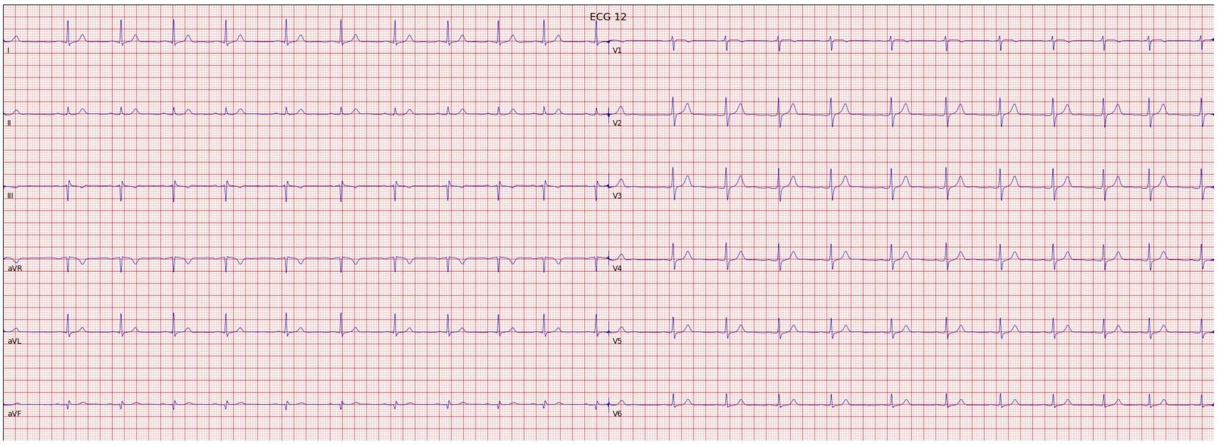


Figure 4.5: Synthetic ECG generated by conditional diffusion model - trained on open source dataset.

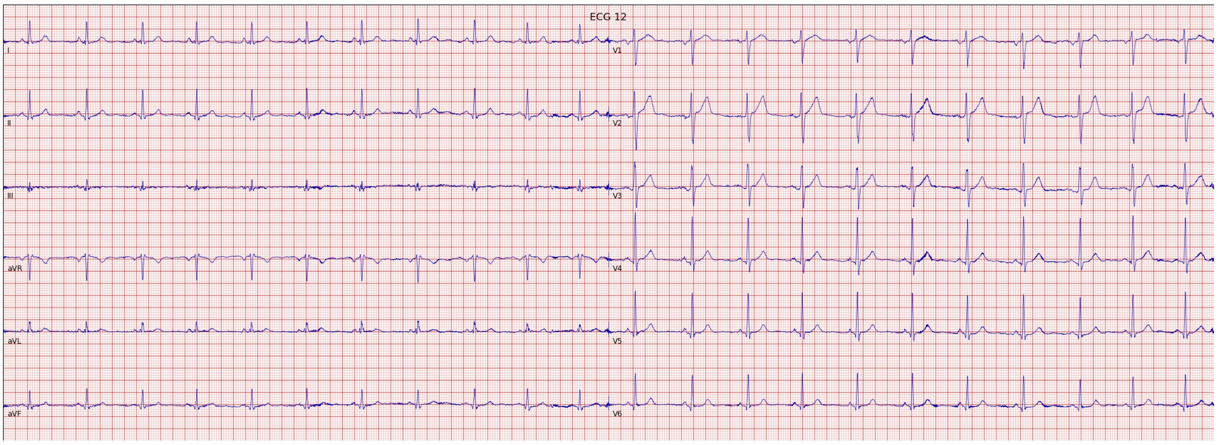


Figure 4.6: Synthetic ECG generated by normal GAN model *AdvP2P-AutoEmb* - trained on private dataset.

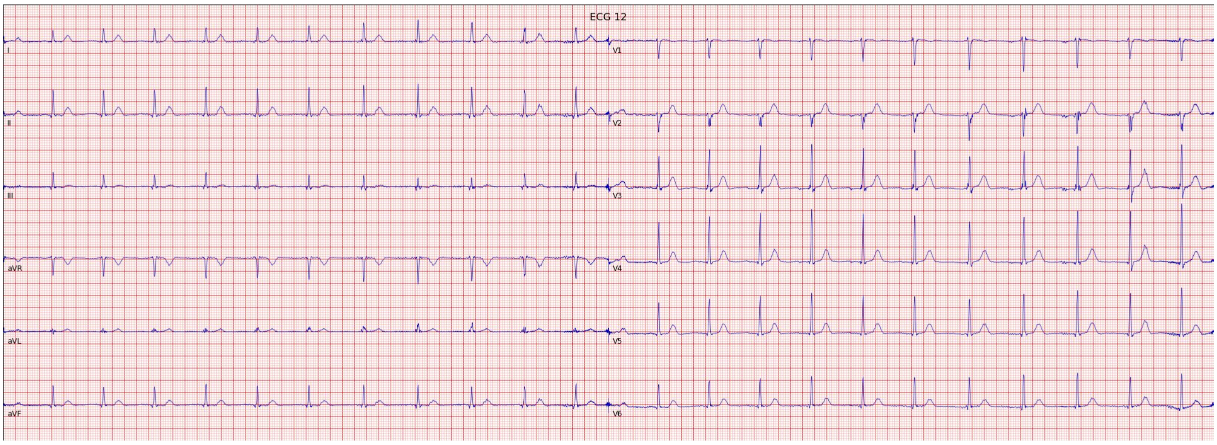


Figure 4.7: Synthetic ECG generated by normal GAN model *AdvP2P-PosEmb* - trained on open source dataset.

better realistic samples or not. We will check different statistics to find the answer to these questions in the sections below.

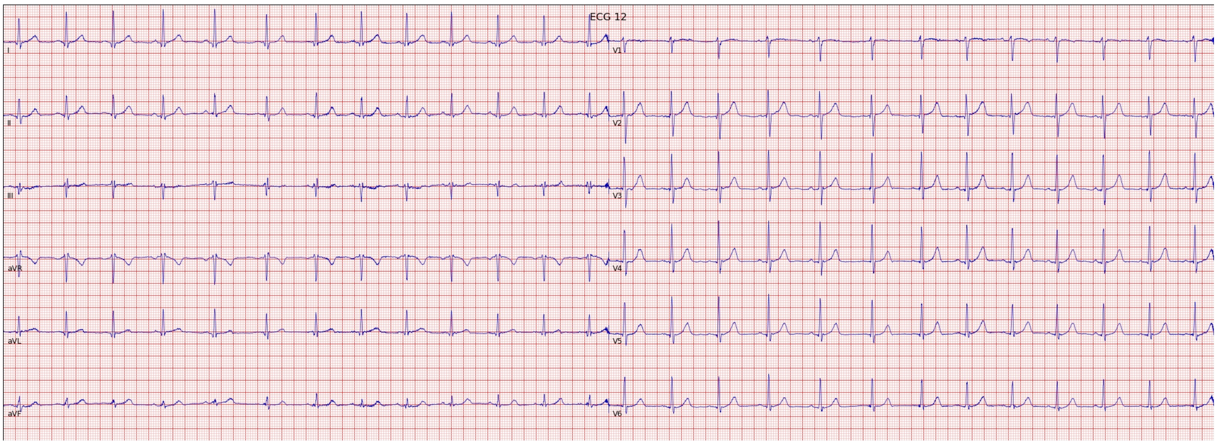


Figure 4.8: Synthetic ECG generated by conditional GAN model *AdvP2P-NtF-AutoEmb* - trained on private dataset.

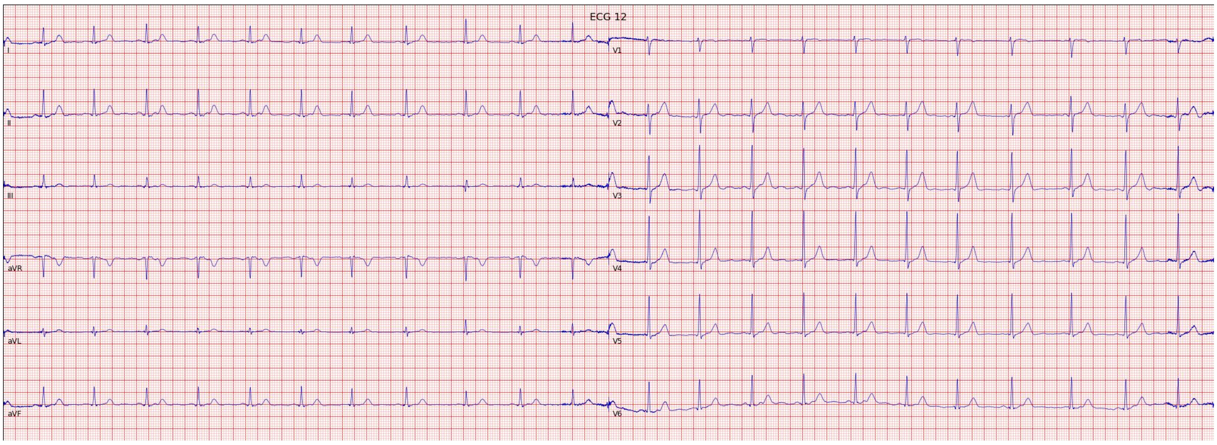


Figure 4.9: Synthetic ECG generated by conditional GAN model *AdvP2P-AutoEmb* - trained on private dataset.

4.5 Similarity Metrics

We are calculating two different similarity evaluation metrics called Inception Score (IS) and Fréchet Inception Distance (FID) which are mostly widely used for evaluating the quality of synthetic data. These two metrics are widely used in the domain of image analysis; however, researcher has used these techniques for evaluating time series data, audio signals, etc. We recommend reading the methodology section to know more about how we calculated these two metrics and what kind of information reveals by these two metrics. Similarity metrics of different models trained on PTB-XL dataset under two different training setting is presented in Table 4.2.

For the sake of clarity, the column header "Normal" refers to models trained without any conditional information. According to Table 4.2, the FID score is highest in diffusion-based model on both training setting, in fact, FID score of diffusion models is almost five time higher than GAN models. This reveals that synthetic ECG generated by GAN based models are closer (realistic) with real ECG samples. Another interesting fact is that, regardless of model types, models trained with conditional information yields relatively smaller FID score. In terms of IS score, there is not much difference between different models, however, the highest (1.13) IS score is yields by conditional GAN based model *AdvP2P-AutoEmb*. Last but not least, our proposed base GAN model (*AdvP2P* which is based in Bi-TCN) and improved version by adding embedding layers interestingly yields smaller FID score than normal TCN based model (*AdvP2P-SD*). Moreover, if we compare the FID score of models based on two embeddings, the model with self-learned embedding layer (*AdvP2P-AutoEmb*) generates the fake ECG with lower score on both type of training.

		Normal		Conditional	
Model	**Types**	**IS↑**	**FID↓**	**IS↑**	**FID↓**
ECG-DDM	Diffusion	1.11	18.41	1.09	16.65
AdvP2P-SD	GAN	1.10	8.11	1.11	5.46
AdvP2P	GAN	1.12	4.87	1.11	4.46
AdvP2P-PosEmb	GAN	1.12	4.07	1.10	4.82
AdvP2P-AutoEmb	GAN	1.11	3.98	**1.13**	**3.36**
AdvP2P-NtF-AutoEmb	GAN	1.11	3.62	1.12	4.52

Table 4.2: IS and FID scores of normal and conditional models trained on PTB-XL dataset. (normal):the GAN (AdvP2P-Auto) model yield the lowest FID score of 3.98 while diffusion yields almost 6 times higher FID score (18.41). (conditional):the AdvP2P-Auto yield the highest IS score (1.13) and lowest FID scores (3.36) out of all models which is alsmot five times smaller than diffusion model. (general): Bi-TCN based models (AdvP2P) yields better result than normal TCN (AdvP2P-SD). AdvP2P2 model is further improved by adding embedding layers (AdvP2P-PosEmb, AdvP2P-AutoEmb).

Similar to the previous Table 4.2, Table 4.3 demonstrates the similarity metrics result of different models trained on private dataset (INT99 and GENSUS). Interestingly, we can see that a similar kind of result pattern exists in this dataset as well compared to open-source dataset result. FID score of diffusion is more than two times higher than GAN models. However, interestingly diffusion model IS is higher than GAN models on both training settings which indicates that diffusion-based model is generating wider variety of ECG. As expected, base GAN model and its variant (which uses bi-TCN) performs better than normal TCN based model (*AdvP2P-SD*). Interesting, in both training situations, *AdvP2P-AutoEmb* yields lowest FID score 4.38 (normal) and 4.73 (conditional). Another interesting thing we can observe in the table is that FID scores of conditional GAN models are slightly higher than normal models. This could be the reason that adding conditional layers increases the complexity in models. Reaching the conclusion (i.e., conditional models are weaker) by looking only FID score may not be the best approach, therefore, we will verify this using ML classifier metrics as well in below sections.

4.6 ML Classifier Metrics

Using machine learning classifier to distinguish real and fake samples are more advance technique. Indeed, it becomes compulsory in analyzing large size of synthetic data. In this section, we have used the same data that we used for calculating similarity metrics. We trained the ML classifier (resnet50) model with fake and real labels. The classification metrics presented in Table 4.4 is result from open-source dataset.

The down arrow in the table header refers that getting smaller value if the best scenario. As you can see in Table 4.4, there is not much difference between normal diffusion and conditional

		Normal		Conditional	
Model	**Types**	**IS↑**	**FID↓**	**IS↑**	**FID↓**
ECG-DDM	Diffusion	1.09	12.36	1.09	10.97
AdvP2P-SD	GAN	1.08	5.27		
AdvP2P	GAN	1.08	4.70		
AdvP2P-PosEmb	GAN	1.08	5.01		
AdvP2P-AutoEmb	GAN	1.08	4.38	1.07	4.73
AdvP2P-NtF-AutoEmb	GAN	1.08	5.00	1.08	5.04

Table 4.3: IS and FID scores of normal and conditional models trained on private dataset (INT99+GENSUS) results. The AdvP2P-AutoEmb model yield the lowest FID distance in normal (4.38) and conditional (4.73) training settings while diffusion models have highest FID distance. Conditional diffusion model improved after adding conditional input while GANs models almost similar FID distance.

		Normal			Conditional		
Model	**Types**	**AC↓**	**PS↓**	**RS↓**	**AC↓**	**PS↓**	**RS↓**
ECG-DDM	Diffusion	0.91	0.95	0.88	0.92	0.95	0.90
AdvP2P-SD	GAN	0.95	0.93	0.96	0.97	0.97	0.97
AdvP2P	GAN	0.96	0.97	0.96	0.91	0.91	0.91
AdvP2P-PosEmb	GAN	**0.81**	0.85	0.79	0.94	0.94	0.94
AdvP2P-AutoEmb	GAN	0.83	**0.84**	0.83	**0.87**	**0.86**	**0.87**
AdvP2P-NtF-AutoEmb	GAN	0.82	0.91	**0.76**	0.91	0.92	0.91

Table 4.4: Result of classification metrics of open-source dataset. Header details: (AC)-Accuracy, (PS)-Precision Score, (RS)-Recall Score. (conditional):ML find easy to distinguish real ECGs and fake ECGs generated by diffusion models (accuracy:0.92) while difficult to distinguish real and fake samples generated by GANs model (accuracy:0.87). However, it is even more difficult for ML model to distinguish normal GANs model samples (accuracy:0.81).

diffusion models. The normal diffusion model achieved 91% of accuracy while conditional got 92%. However, both of the models have the same precision score (PS) of 0.95. On the other hand, if we compare the result of diffusion model with best performing GAN model, GAN models (*AdvP2P-PosEmb*) reduced the accuracy from 91% to 81% which is huge improvement. This indicates that it becomes harder for machine learning models to distinguish between real and fake samples generated by *AdvP2P-PosEmb* model. Another interesting observation is that single dimension (normal TCN) based model (*AdvP2P-SD*) performed worst in both normal and conditional training setting, indeed worse than diffusion models. However, our proposed based model yields the worst result (96%) in normal training setting and better in conditional training (91%). On the other hand, interesting, the variant of base model with embedding layers outperforms both diffusion and base model with the accuracy of *AdvP2P-PosEmb*-81%, *AdvP2P-AutoEmb*-83% and *AdvP2P-NtF-AutoEmb*-82%.

A similar pattern exists in the conditional side of models (i.e., embedding base model performs better). The lowest accuracy yields by model is 87% which is given by **AdvP2P-AutoEmb**. But the most important thing to notice here is that as compared to normal models (without any condition), conditional models have higher accuracy values. This indicates that ML model can distinguish the real and fake samples more correctly. We will check whether similar patterns exist in private dataset as well or not before making some decision.

		Normal			Conditional		
Model	**Types**	**AC↓**	**PS↓**	**RS**	**AC↓**	**PS↓**	**RS↓**
ECG-DDM	Diffusion	0.94	0.93	0.93	0.92	0.89	0.93
AdvP2P-SD	GAN	0.86	0.82	0.84			
AdvP2P	GAN	0.89	0.87	0.87			
AdvP2P-PosEmb	GAN	0.77	0.60	0.80			
AdvP2P-AutoEmb	GAN	0.75	0.66	0.72	0.82	0.71	0.84
AdvP2P-NtF-AutoEmb	GAN	0.75	0.57	0.76	0.79	0.69	0.78

Table 4.5: Result of classification metrics of private dataset. Header details: (AC)-Accuracy, (PS)-Precision Score, (RS)-Recall Score. (conditional):ML model has low error on distinguish real samples and fake samples generated by diffusion model (accuracy:0.92) while higher errors in distinguish real samples and fake samples generated by GANs (accuracy:0.79).

The result presented in Table 4.5 demonstrate the ML classifier metric from private dataset. If we closely observe the result, there are two major differences as compared to result presented in table 4.4. First, adding conditional information improves the quality of generated samples as ML accuracy decreases from 94 to 92%. Second, all the GAN models outperform diffusion models, i.e., samples generated by GAN models look more realistic in private dataset. Similarly, there are two similarities between the results of two datasets. The first one, as usual, proposed embedding layer-based models yields lower accuracy value. Indeed, the difference (94%-75%=19%) is huge in between diffusion and GAN model (*AdvP2P-AutoEmb GAN*). The second one is that conditional model outputs are classified with higher accuracy as in open-source dataset. This indicates that adding conditional information increases the complexity in GAN models.

The optimal goal is to fool the classifier, i.e., let the classifier classify fake as real and real as fake. In that scenario, embedding based GAN models do an excellent job. For example, model *AdvP2P-NtF-AutoEmb* yields precision score of 0.57 which indicates that almost 43% of fake sample exits in test dataset are also classified as real. In the case of conditional model, the score is 0.69 which is awesome as compared with other models.

4.7 Power Spectrum Analysis (PSA)

Power spectrum analysis is a key tool that we have used for analyzing fake and real samples. Based on similarity metrics and ML classifier metrics, we can see that diffusion models are not able to generate realistic ECG signal as compared to GAN. But we cannot say why or where there is a problem in sample generated by diffusion model. PSA analysis provides more insights of different frequencies exits in the ECG and their strength so that we can compare real and fake. As the result shows that both datasets have similar patterns, we are only using private datasets for PSA analysis. Power spectrum analysis of real and fake samples generated by conditional diffusion models are presented in Figure 4.10.

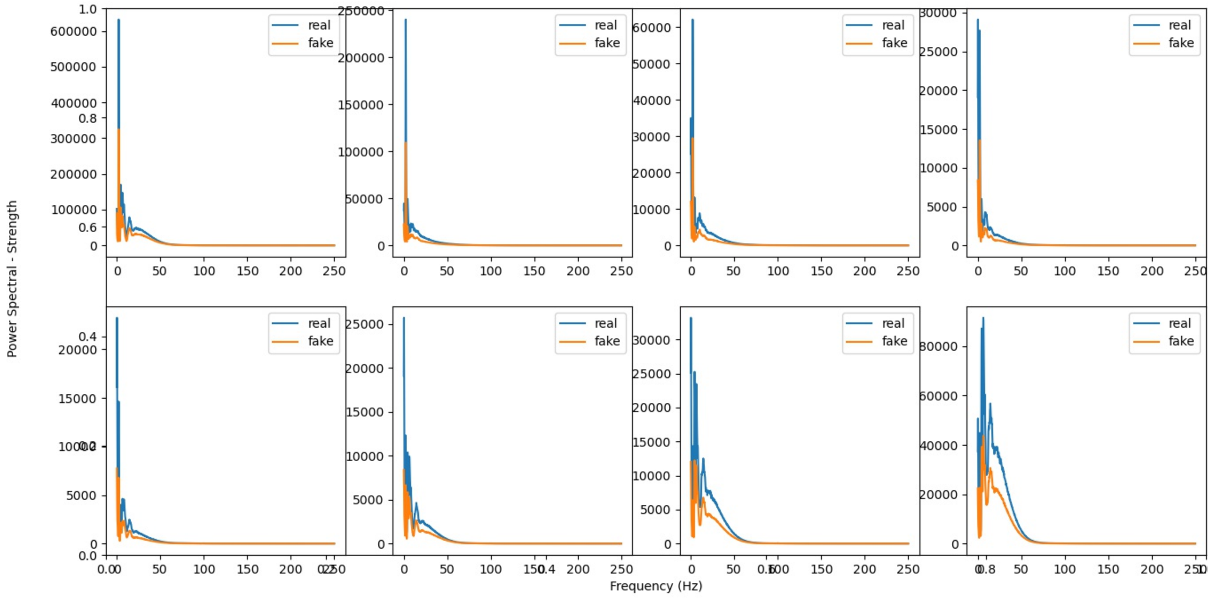


Figure 4.10: Power spectrum analysis of real samples and fake samples generated conditional diffusion model. Mean power spectrum analysis represent that diffusion model capture peaks and waves but their strength are lower than real ECGs. Top four graphs belongs to I, II, V1, and V2 leads. The bottom four graphs belongs to V3, V4, V5, and V6 (from left to right).

The mean value of PSA is compared at lead (channel) level in Figure 4.10. The figure shows that ECG signal has one peak with very high strength whose frequency is around 1/2Hz and some other waves with smaller strength. The frequency with highest strength refers to R peaks wave and other small waves such as P and W wave. According to Figure 4.10, the diffusion-based model is capturing the patterns i.e it generates the different waves in ECG, but they are very weak signal. In every channel, all the frequency are exits as in real and at a same pattern i.e if real sample has peak at 1Hz, fake also have peak at that frequency which is good, but the problem is that fake samples has relatively lower power on each frequency which will have direct impact on waves. Fake samples also have different waves (p, w, p) but they are small in nature.

As shown in Figure 4.10, similar compassion is made between real and fake samples generated by conditional GAN model *AdvP2P-NtF-AutoEmb* which is presented in Figure 4.11. The PSA comparison is promising as compared to diffusion model's samples. As we can see, over a range of frequencies, the mean value of strength is overlapping which is interesting. In some of the leads, for example, first two (I, II), firth (v3) and last (v6) leads, the average strength is almost similar. This represents that all the waves that exist in real samples have been generated by the model. However, if we observe the middle two graphs (i.e lead V4 and

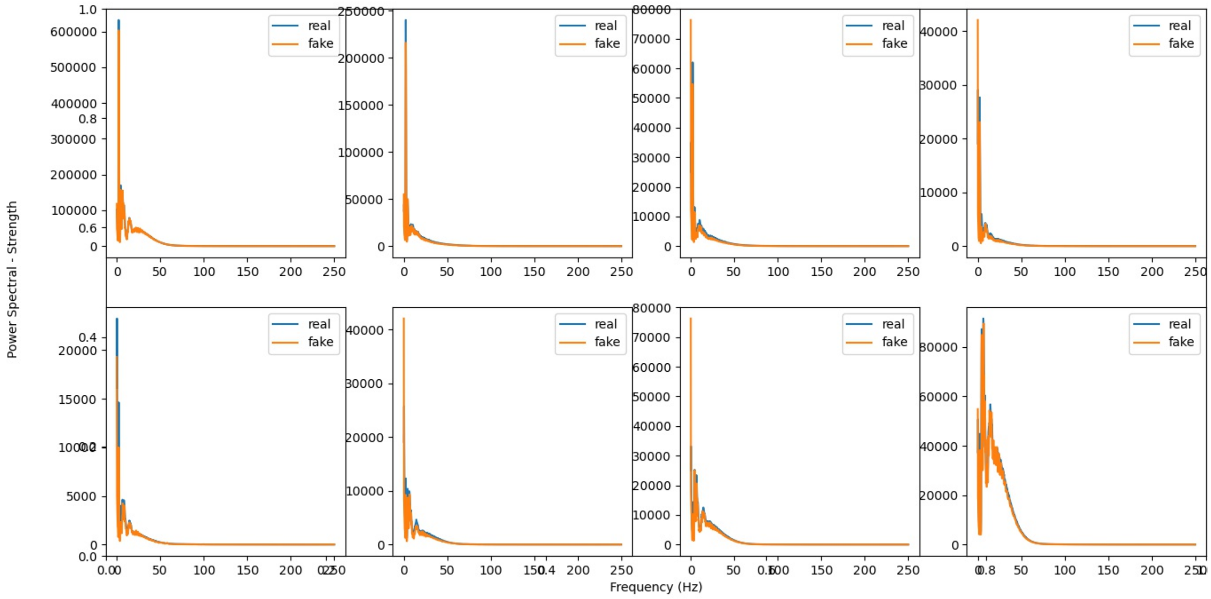


Figure 4.11: Power spectrum analysis of real samples and fake samples generated by conditional GAN (*AdvP2P-NtF-AutoEmb*) model.Mean power spectrum analysis represent that the conditional GAN (*AdvP2P-NtF-AutoEmb*) model not only capture peaks and waves but their strength are also similar to real ECGs. Top four graphs belongs to I, II, V1, and V2 leads. The bottom four graphs belongs to V3, V4, V5, and V6 (from left to right). (issue):In some leads (V1, V2, V4, V5), at highest peaks, the strength of fake samples are higher than normal.

V6) in the bottom layer of figure, the generated fake samples have higher strength than real ones which means model are generating higher peaks in those channels. We think this is one of the reasons why ML model classifies the real and fake with an accuracy of 75% and a precision score of 0.57.

4.8 ECG life cycle

In the previous power spectrum analysis, we saw that diffusion-based models have weaker (specially R-peaks) signals while GAN based models generate signals similar to real. To further confirm that weak/good signals are presented in ECG signals, we are analyzing one life cycle of ECG which starts before p-waves to end after w-wave. For this analysis, we choose the time window of 1500 to 3000 points, and find out the highest peaks between these two time points, and finally extract 200 points before and after the peak point. In total, we extract 400 points from each channel and compare the result at channel level. Figure 4.12 demonstrates one ECG life cycle of private dataset.

In the figure red, green and blue represent mean value of conditional diffusion, real and conditional *AdvP2P-AutoEmb* signal. Similarly, the shadow part represents the standard deviation differences. The result shows that all models have created the ECG life cycle which is promising. However, if we closely observe, we can see R-peaks generated by diffusion model are weaker in every lead. Similarly, we can observe the noticeable difference in w-wave of some of the channels (leads: V1, V2, V3, V4). On the other hand, signals generated by GAN model are identical with real signals. One interesting difference that we observe in between real and GAN signals is that at R-peaks, the standard deviation value of GAN signals is higher which indicates that GAN models have higher deviation in generating R-peaks.

As compared to private dataset, ECG life cycle of open-source dataset presented in Figure

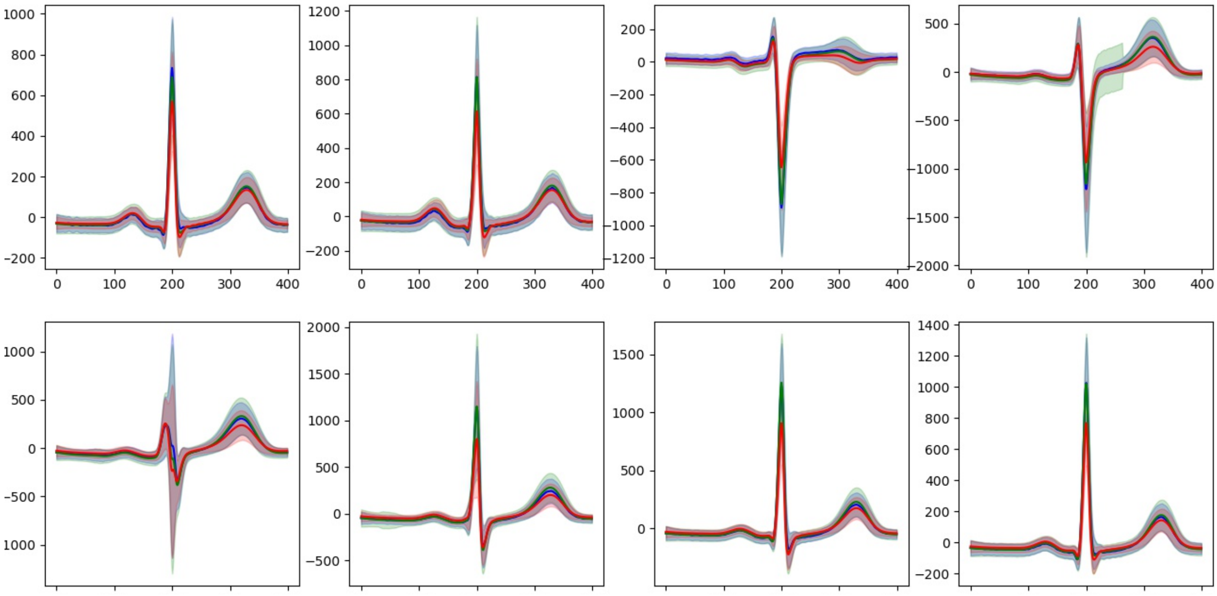


Figure 4.12: Comparison of ECG life cycle of real and generated fake samples of private dataset. Signal are represented by color where Green:real, Blue:conditional *AdvP2P-AutoEmb*, Red:conditional *ECG-DDM*. Top four graphs belongs to I, II, V1, and V2 leads. The bottom four graphs belongs to V3, V4, V5, and V6 (from left to right). Real and fake ECGs generated by conditonal GAN are very similar while Diffusion has noticeable issue in R preaks and W-waves.

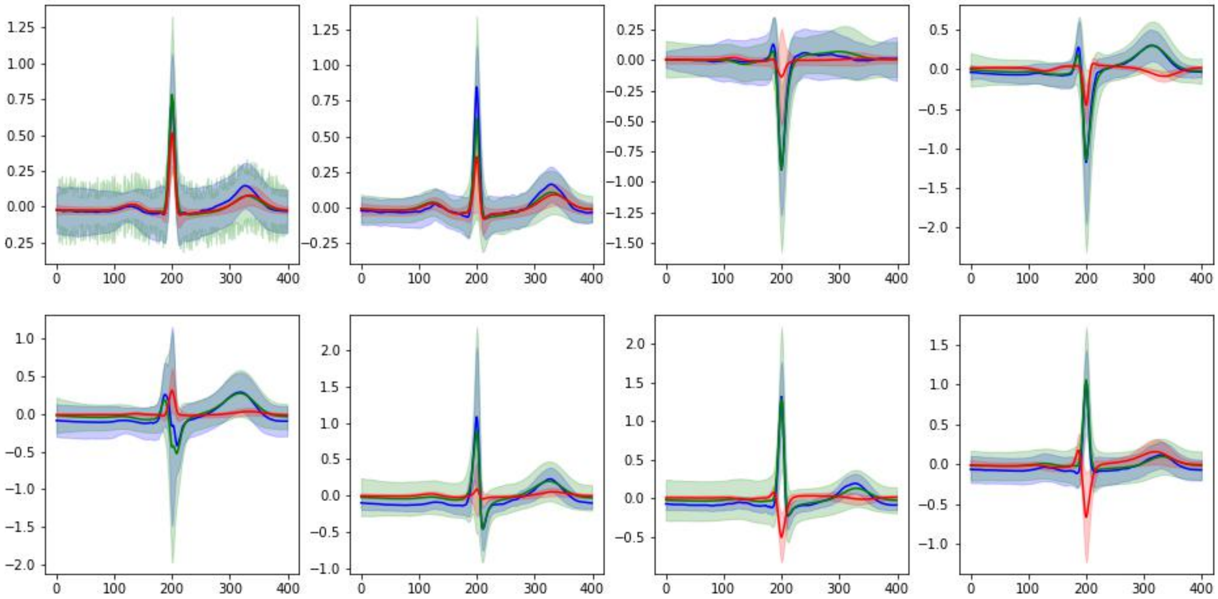


Figure 4.13: Comparison of ECG life cycle of real and generated fake samples of open-source dataset (PTB-XL). Green:real, Blue:conditional *AdvP2P-AutoEmb*, Red:conditional *ECG-DDM*. Top four graphs belongs to I, II, V1, and V2 leads. The bottom four graphs belongs to V3, V4, V5, and V6 (from left to right). Real and fake ECGs generated by conditonal GAN are very similar while Diffusion has worst signal in most of leads except first two leads (I, II).

4.13 reveals completely unexpected hidden truth. In channels V1, V2, V3, and V4, diffusion models generate very small R-peaks while completely failing to generate W-wave. Similarly, in the last two channels (v5 and V6), in the real and GAN's signals have R-peaks at positive side while diffusion has a negative side. There could be two possible reasons behind this type of unexpected result. The first one is higher diversity in ECG signals makes model different to learn Gaussian noise to input, and the second one is Gaussian noise distribution that we are passing for generating fake ECG signals. More in-depth analysis and comparison of noise needs to be done in future work.

4.9 Latent Space Visualization

Latent space visualization gives the ability to check how the distribution of real and fake samples looks like in lower dimension space. We have explained more about the approach in the result evaluation section of methodology chapter. Latent space visualization of real samples with samples from diffusion model and GAN model under normal and conditional setting is presented in Figure 4.14.

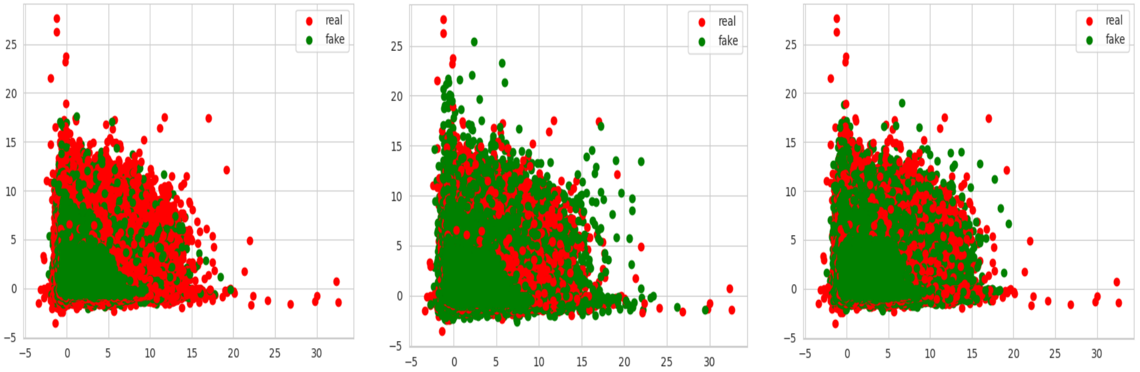


Figure 4.14: Latent space visualization - (Left): fake samples by diffusion, (Middle): fake samples by *AdvP2P-NtF-AutoEmb*, (Right): fake samples by Conditional *AdvP2P-NtF-AutoEmb*.

Above Figure 4.14 shows another interesting insight as compared to previous analysis which is the issue of diversity. The first graph in the figure is the comparison between real and fake samples generated by diffusion model. In the graph, we can observe that two distributions overlap with each other, however, fake samples are centered in one location and do not capture the pattern of real distribution. This could be the reason that diffusion-based models are generating similar types of ECG most of the time. In the domain of generative models, this type of issue is called diversity issue. On the other hand, GAN based model *AdvP2P-NtF-AutoEmb* under both training setting are quite good at capturing the different patterns exits in real samples. However, if we compare both closely, middle graph, i.e., GAN trained without any condition looks better in capturing pattern. If we compare the analysis that we presented in previous section of ML classifier metrics, i.e., hiving higher value of recall score could be the issue of diversity (normal-recall score 0.76, conditional-recall score 0.78), this statement is further justified by latent space visualization presented in Figure 4.14.

4.10 MUSE Feature Analysis

This is one of the key important analyses of this thesis work. There are two important questions that we are going to answer from this analysis. They are 1) do the fake samples have similar ground truth as reals? and 2) Do the models have learned ground truth correctly so that it generates the fake sample of given ground truth? To answer these two questions, we have performed two types of analysis. The first one is at population level which will help to answer whether adding conditional information improves the model to generate required distribution or not. Similarly, at the second approach, we compare at one-to-one level where we pass the same defined ground truth to GAN and Diffusion model and compare which one generates closer ground truth. MUSE analysis is only performed on private dataset (Int99 and GENSUS). There are two reasons behind this. The real MUSE tool was very busy with another test, we were kept in queue until the end of thesis. As an alternative we are using AI based ECG parameters extractor [38] trained on same private dataset. The AI-ECG parameter extractor is not available for open source PTB-XL dataset.

4.10.1 Population Level Analysis

For the consist result comparison, we have used the same number of samples (7077) from each group. For this analysis, we first extract the ground truth from training ECG samples using the AI-based model. Then, we extracted the ground truth from non-conditional (normal) diffusion model. For conditional diffusion model, we passed the training samples ground truth as conditional information and extracted the ground truth from generated sample so that we ca make comparison. The Figure 4.15 demonstrates the heart rate distribution comparison and heart rate relationship with QT interval.

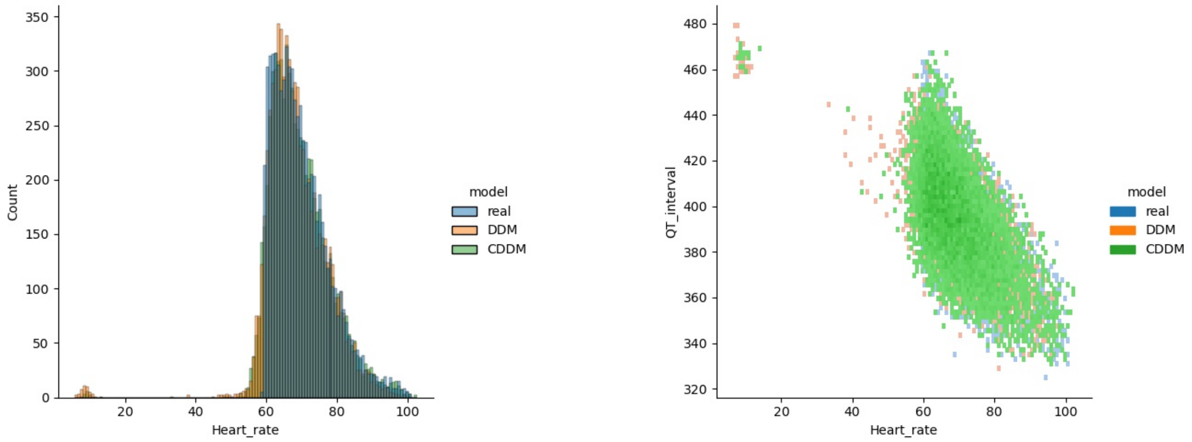


Figure 4.15: Diffusion models result comparison with real ground truth of private dataset (INT99, GENSUS). (Left): heart rate distribution comparison, (Right): heart rate versus QT interval relationship.

The left graph on Figure 4.15 shows the comparison of synthetic heart rate of conditional and on-conditional diffusion model with the real heart rate. At the glance, we can see that the normal heart rate (60-100) distribution are overlapping and identical. However, we can observe that both diffusion models have also generated an extremely low heart rate which does not fall under normal heart rate range. On the other hand, in general, there is a strong correlation between heart rate and QT interval. The graph on the right side of Figure 4.15 demonstrates correlation between real and synthetic ECG. The correlation between heart rate and QT interval of conditional diffusion model is overlapping. If we carefully observe, we can see that conditional model is a bit better than normal. Interestingly, QT interval of fake samples belongs to normal range regardless of low heart rate.

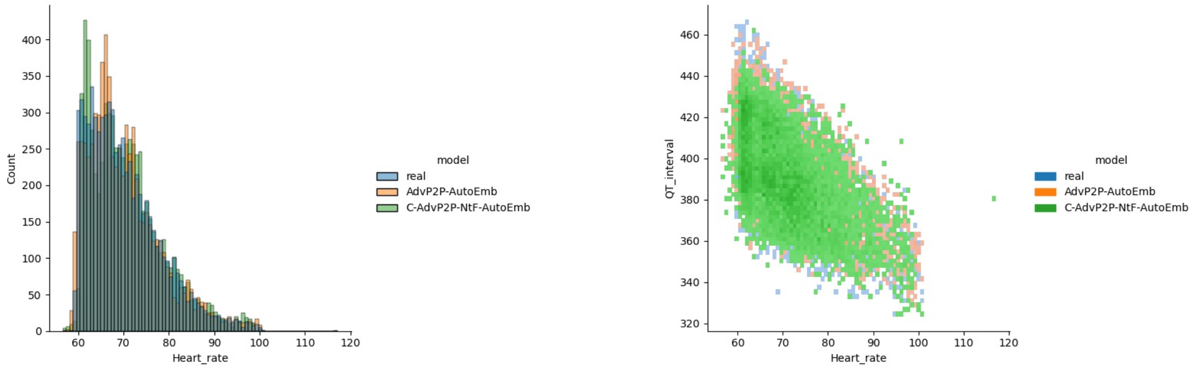


Figure 4.16: GAN models result comparison with real ground truth of private dataset (INT99, GENSUS). (Left): heart rate distribution comparison, (Right): heart rate versus QT interval relationship.

As the result presented in Figure 4.15, similar result is presented in Figure 4.16 based on GAN model. For these experiments, we have selected *AdvP2P-AutoEmb* as non-conditional GAN and *AdvP2p-NtF-AutoEmb* is used from conditional GAN. In the figure letter 'C' before model name refers to conditional. The heart rate of conditional and non-conditional GAN falls under the range of real heart rate. As compared to diffusion models, one interesting noticeable difference is that the model has generated a high number of ECG with the most common heart rate (around 70 and around 65). This is the common issue of GANs, i.e., the issue of diversity. Similarly, the heart rate correlation with QT interval presented on right side of Figure 4.16 demonstrates that both conditional and non-conditional model have similar correlation as real. If we observe closely, we can see that both of the models have issues in generating the higher QT interval.

Based on only the result presented in Figures 4.15 and 4.16, we cannot figure out that adding conditional model improves better ECG generation. Therefore, we have also performed an analysis based on the measurement of four different parameters. The mean, standard deviation, 2.5% and 97.5% percentile difference of seven different ECG parameters (ground truth) of real and fake samples are presented in Table 4.6. In the table, the first four parameters (Heart Rate, PR interval, QRS duration and QT interval) are also used as conditional while the last three parameters (STJ, Rpeak and Tpeak) are not part of given condition to model however these parameters are important to figure out the effect of conditional parameter on non-conditional parameters.

		Real-normal ()				DDM-normal (10,000)				CDDM-normal (10,000)			
		Mean	Std	2.5%	97.5%	Mean	Std	2.5%	97.5%	Mean	Std	2.5%	97.5%
Heart rate	BPM	70	8	60	90	69	9	57	87	70	9	58	90
PR interval	ms	156	19	120	196	156	18	124	192	157	19	122	195
QRS duration	ms	89	9	72	108	85	7	73	102	90	9	75	108
QT interval	ms	395	21	352	436	396	20	357	435	395	22	352	438
STJ	μV	3	26	-45	59	-3	20	-39	42	-5	21	-44	41
R peak	μV	1295	420	585	2208	895	351	338	1727	933	352	380	1768
T peak	μV	346	136	132	668	254	111	91	520	270	111	105	533

Table 4.6: Mean, standard deviation (std), 2.5%, and 97.5% percentile comparison of ground truth from real ECGs and fake ECGs generated by diffusion models.

In overall, the mean, standard deviation, 2.5% and 97.5% percentile of different ECG parameters measured from real, and fake presented in Table 4.6 demonstrates that adding conditional parameters to the diffusion model generate samples with more realistic ECG parameters. More specially, we can observe the huge improvement in QRS duration where the mean value of QRS duration in original is 89 while non-conditional model has 85 but the conditional model has 90. Another interesting thing to observe is that though R peak and T peaks were not part of conditional parameters, adding other conditional parameters has also improved the R peaks and T peaks however they are still not close to normal ECG parameters. This is (i.e., weak R peaks in diffusion model) one of the analyses that we made based on PSA analysis and ECG life cycle before which is now further proved from this analysis.

Table 4.7 compare the results generated by GAN models. Adding the ground truth to the model perfectly generates the mean PR interval of 156 by conditional model while 154 by non-conditional model. Similarly, non-conditional GAN model generates a very high T peak of 383 while the real T peak is 346. The conditional model generates a much closer mean T peak of 351. On the other hand, there is no improvement in the case of QRS duration and QT interval.

		Real-normal (7077)				AdvP2P-AutoEmb (10,000)				C-AdvP2P-NtF-AutoEmb (10,000)			
		Mean	Std	2.5%	97.5%	Mean	Std	2.5%	97.5%	Mean	Std	2.5%	97.5%
Heart rate	BPM	70	8	60	90	70	8	60	91	71	8	61	91
PR interval	ms	156	19	120	196	154	14	128	182	156	11	137	178
QRS duration	ms	89	9	72	108	93	7	80	109	93	6	81	107
QT interval	ms	395	21	352	436	398	20	356	434	390	21	352	429
STJ	μV	3	26	-45	59	9	31	-49	75	12	35	-54	84
R peak	μV	1295	420	585	2208	1316	398	608	2157	1326	374	675	2059
T peak	μV	346	136	132	668	382	134	157	671	351	112	174	597

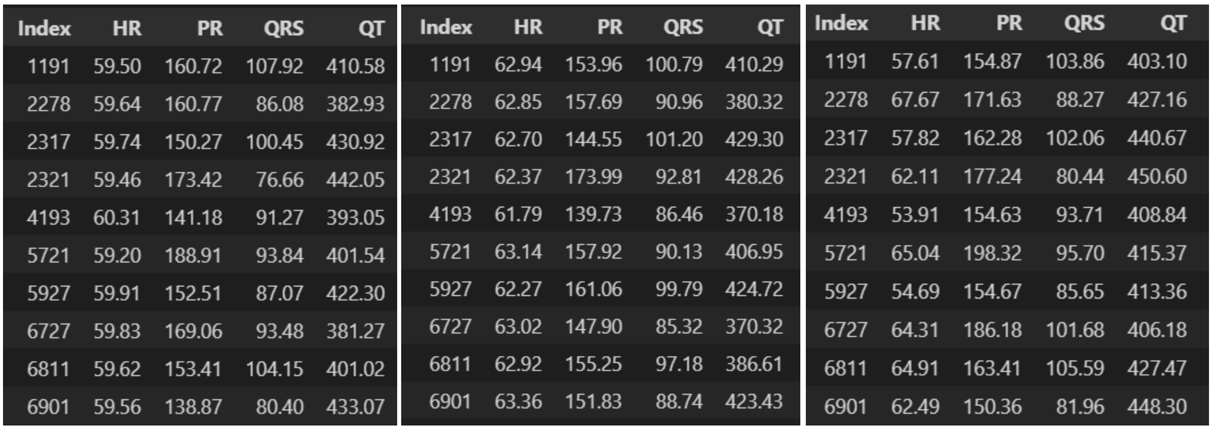
Table 4.7: Mean, standard deviation (std), 2.5%, and 97.5% percentile comparison of ground truth from real ECGs and fake ECGs generated by GAN models.

4.10.2 One to One Analysis

From the population level analysis, we can observe that adding conditional information to the model during the generation has improved some for the parameter of ground truth. However, the analysis is done at population level so that we cannot confirm that when we pass the ground truth, ECG signal of similar ground truth is generated or not. This is very important to know which helps to make decision that whether model has mapped the relation of condition parameter without or not. Therefore, we have made three different scenarios: 1) low heart rate (<60), 2) high heart rate (>95), and 3) medium heart rate (66-72). For each case, we select ten different ground truths and pass the same ground truth to GAN and diffusion model so that we can also answer which model is better at mapping condition to output. We have passed 7 parameters as input but as we have only four parameter ground truth, we are only comparing four parameters. They are HR-Heart Rate, PR-PR interval, QRS-QRS duration, and QT-QT interval.

The Figure 4.17 demonstrates the comparison of ECG parameters generated from GAN and diffusion models for the same given ground truth. The result shows that GAN based models are comparatively closer and consistent to real ground truth while diffusion-based models are quite varied (high-67.67 and low-53.91). In terms of other parameters, GAN is also comparatively better such as PR interval.

The result of the second scenario (HR>95) is presented in Figure 4.18. As usual, conditional GAN result is very impressive and closer to the given ground truth as compared to conditional diffusion model. The diffusion model has also generated unrealistic parameters as well, for example, index 6668 which has HR of 97.35 but the diffusion model generates 40.39 which is completely out of normal heart rate. However, interestingly, we observe that diffusion model is at learning and mapping PR and QRS duration as compared to GAN.

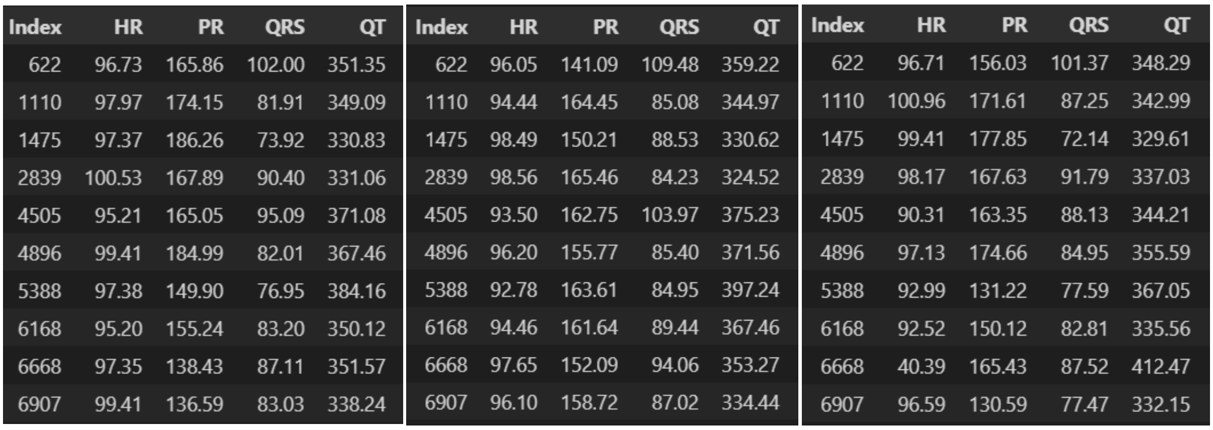


Index	HR	PR	QRS	QT
1191	59.50	160.72	107.92	410.58
2278	59.64	160.77	86.08	382.93
2317	59.74	150.27	100.45	430.92
2321	59.46	173.42	76.66	442.05
4193	60.31	141.18	91.27	393.05
5721	59.20	188.91	93.84	401.54
5927	59.91	152.51	87.07	422.30
6727	59.83	169.06	93.48	381.27
6811	59.62	153.41	104.15	401.02
6901	59.56	138.87	80.40	433.07

Index	HR	PR	QRS	QT
1191	62.94	153.96	100.79	410.29
2278	62.85	157.69	90.96	380.32
2317	62.70	144.55	101.20	429.30
2321	62.37	173.99	92.81	428.26
4193	61.79	139.73	86.46	370.18
5721	63.14	157.92	90.13	406.95
5927	62.27	161.06	99.79	424.72
6727	63.02	147.90	85.32	370.32
6811	62.92	155.25	97.18	386.61
6901	63.36	151.83	88.74	423.43

Index	HR	PR	QRS	QT
1191	57.61	154.87	103.86	403.10
2278	67.67	171.63	88.27	427.16
2317	57.82	162.28	102.06	440.67
2321	62.11	177.24	80.44	450.60
4193	53.91	154.63	93.71	408.84
5721	65.04	198.32	95.70	415.37
5927	54.69	154.67	85.65	413.36
6727	64.31	186.18	101.68	406.18
6811	64.91	163.41	105.59	427.47
6901	62.49	150.36	81.96	448.30

Figure 4.17: ECG ground truth with lower heart rate (>60). Left: Real ground truth passed in GAN and Diffusion model, Middle: Features generated by GAN model for given ground truth, Right: Features generated by diffusion model for given ground truth.

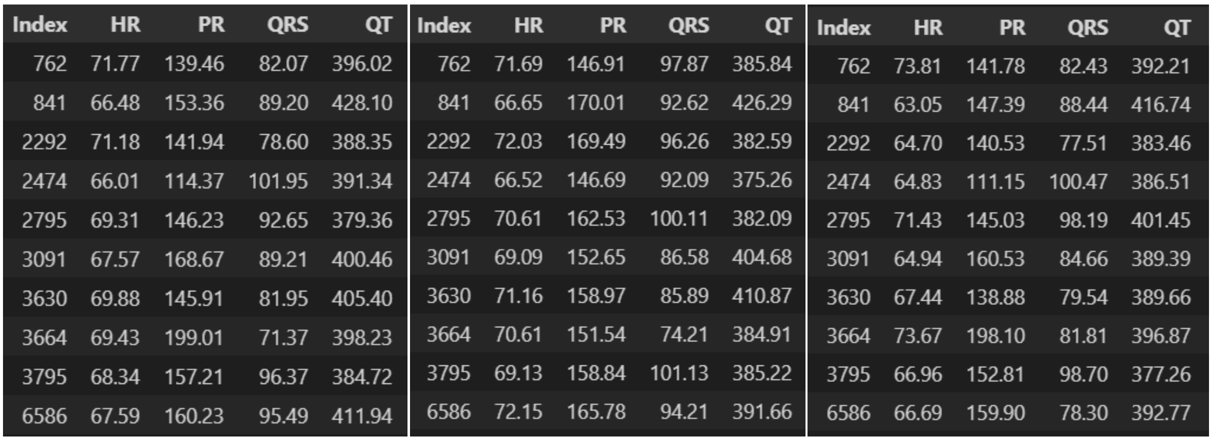


Index	HR	PR	QRS	QT
622	96.73	165.86	102.00	351.35
1110	97.97	174.15	81.91	349.09
1475	97.37	186.26	73.92	330.83
2839	100.53	167.89	90.40	331.06
4505	95.21	165.05	95.09	371.08
4896	99.41	184.99	82.01	367.46
5388	97.38	149.90	76.95	384.16
6168	95.20	155.24	83.20	350.12
6668	97.35	138.43	87.11	351.57
6907	99.41	136.59	83.03	338.24

Index	HR	PR	QRS	QT
622	96.05	141.09	109.48	359.22
1110	94.44	164.45	85.08	344.97
1475	98.49	150.21	88.53	330.62
2839	98.56	165.46	84.23	324.52
4505	93.50	162.75	103.97	375.23
4896	96.20	155.77	85.40	371.56
5388	92.78	163.61	84.95	397.24
6168	94.46	161.64	89.44	367.46
6668	97.65	152.09	94.06	353.27
6907	96.10	158.72	87.02	334.44

Index	HR	PR	QRS	QT
622	96.71	156.03	101.37	348.29
1110	100.96	171.61	87.25	342.99
1475	99.41	177.85	72.14	329.61
2839	98.17	167.63	91.79	337.03
4505	90.31	163.35	88.13	344.21
4896	97.13	174.66	84.95	355.59
5388	92.99	131.22	77.59	367.05
6168	92.52	150.12	82.81	335.56
6668	40.39	165.43	87.52	412.47
6907	96.59	130.59	77.47	332.15

Figure 4.18: ECG ground truth with higher heart rate (>95). Left: ground truth used as condition, Middle: Features generated by GAN model for given ground truth, Right: Features generated by diffusion model for the same given ground truth.



Index	HR	PR	QRS	QT
762	71.77	139.46	82.07	396.02
841	66.48	153.36	89.20	428.10
2292	71.18	141.94	78.60	388.35
2474	66.01	114.37	101.95	391.34
2795	69.31	146.23	92.65	379.36
3091	67.57	168.67	89.21	400.46
3630	69.88	145.91	81.95	405.40
3664	69.43	199.01	71.37	398.23
3795	68.34	157.21	96.37	384.72
6586	67.59	160.23	95.49	411.94

Index	HR	PR	QRS	QT
762	71.69	146.91	97.87	385.84
841	66.65	170.01	92.62	426.29
2292	72.03	169.49	96.26	382.59
2474	66.52	146.69	92.09	375.26
2795	70.61	162.53	100.11	382.09
3091	69.09	152.65	86.58	404.68
3630	71.16	158.97	85.89	410.87
3664	70.61	151.54	74.21	384.91
3795	69.13	158.84	101.13	385.22
6586	72.15	165.78	94.21	391.66

Index	HR	PR	QRS	QT
762	73.81	141.78	82.43	392.21
841	63.05	147.39	88.44	416.74
2292	64.70	140.53	77.51	383.46
2474	64.83	111.15	100.47	386.51
2795	71.43	145.03	98.19	401.45
3091	64.94	160.53	84.66	389.39
3630	67.44	138.88	79.54	389.66
3664	73.67	198.10	81.81	396.87
3795	66.96	152.81	98.70	377.26
6586	66.69	159.90	78.30	392.77

Figure 4.19: ECG ground truth with medium heart rate (66-72).Left: Real ground truth passed in GAN and Diffusion model, Middle: Features generated by GAN model for given ground truth, Right: Features generated by diffusion model for given ground truth.

The third scenario, i.e., ground truth with the most common heart rate (66-72) result is presented in Figure 4.19. Compared to the previous two results, we can observe that conditional GAN model is more accurate in mapping the ground truth in the generated ECG. For example, heart rates at the index 762, 841, 2474, etc. are almost the same in generated ECG as well. On the other hand, conditional diffusion model has relatively higher loss on heart rate. However, interestingly, this time conditional diffusion model also has excellent results in mapping QT interval and QRS duration.

In overall, there are some interesting findings that we can take away from this one-to-one analysis. First, though there is minor difference in the real and generated ground truth, the model can map the condition into output. The generated ground truth distribution is nearby but not completely out of distribution. Second, GAN model gives an impressive result for majority sample groups (66-72) and vice versa for minority sample group (<60 and >95). This is an expected and common issue of GAN model. Third, the diffusion model is excellent in mapping QT and QRS duration, but sometimes it generates a completely unrealistic ECG, for example, heart rate of 40.

4.11 Expert Review

We have asked doctors to look at the samples generated by different models and provide feedback on the model result. From the eyes of experts, which model result looks good, what kind of information is captured in general, and what kind of limitation exists in the generated fake samples are interesting to present. For this experiment, we randomly generate 100 ECG plots from each model and provide it to doctor/experts to check one by one and provide their feedback. Here is the feedback provided by the doctors.

Overall impression: Most ECGs look like real ECGs at first glance. In most cases, it is quite difficult to distinguish whether the ECGs are real or fake. From the observation of the doctors, they feel that almost 80% of the ECGs look real. However, there are some variations between the models.

Poor model: According to experts, the poor models are diffusion-based models both conditional and non-conditional. Similarly, from the GAN side, ECG generated by the model *AdvP2P-SD* (i.e., the model without bidirectional TCN (Single Direction)) is comparatively weak. Diffusion-based models often have problems with timing. Either a sudden pause or a new beat too fast without change in morphology. The experts mention that unexpected pauses often are a major issue in diffusion models. Similarly, lower amplitude is another noticeable problem in general. Of 100 ECGs, one of the ECGs was black, which indicates that the model completely fails to generate ECGs. On the other hand, *AdvP2P-SD* GAN model often generates ECGs with strange/weird P-waves and T-wave.

Good model: On the other hand, experts found that *AdvP2P-PosEmb* and *C-AdvP2P-NtF-AutoEmb* models from GAN are comparatively and consistently good at generating fake ECG signals. Out of 100 samples, in 2/3 samples experts observe the issue of weird P-wave and T-wave, which is the most common issue in most of the other models.

4.12 Summary

In this section, all proposed models in the methodology chapter 3 are trained on two different datasets. After the completion of model training, data is generated from different checkpoints. Based on similarity scores and classifier metrics, the best model checkpoint is identified and used for further analysis. The generated data are evaluated using different methods defined at section 3.6 of methodology.

The experimental results and analysis presented in this section are summarized in the following paragraphs. Out of two generative models, under both training settings (conditional/unconditional), generative adversarial networks (GAN) models are clear winners in every evaluation method. Similarly, out of different variants of GANs, proposed Bi-TCN based (*AdvP2P*) model is better than the normal TCN based model (*AdvP2P-SD*). The experimental result shows that the embedding layer that we added on top of the proposed Bi-TCN model is even better. All these experimental results answered the first objective of research, i.e., out of two state-of-the-art models, GANs model can generate better quality of fake ECGs signal in unconditional model.

Similarly, the experimental results of the conditional models demonstrate that diffusion-based models are improved compared to nonconditional models based on the statistic of similarity scores and classification metrics. However, the diffusion model is still not better than GANs model. On the other hand, adding conditional information to the GAN models does not contribute to improving the FID and classification metrics. This is reasonable. However, the main objective is to confirm whether the models are generating similar ECGs as inputted in the conditional information or not. We verified this by one-to-one analysis of input ECG parameters and fake ECGs parameters. The analysis demonstrated that models are generating ECGs similar to the given conditional information. Out of two generative models, as usual, the GANs model is better and more consistent in generating conditional ECGs. However, we observed that not all features are perfectly mapped in the generated ECGs. This is maybe due to many parameters passed together. We will discuss more about this in the next chapter of discussion and future work. From this analysis, we can confirm that the main defined research question and our second objective is achieved, i.e., the generative model can generate ECGs signals according to the given conditional information.

Another interesting observation that we observed from the analysis of the generated data is that there is no single standard open-source methods (similarity metrics, classification metrics, latent space visualization, and power spectrum analysis) that can tell the quality of the generated synthetic ECGs. However, the combination of all shows a better picture. On the other hand, professional tools (MUSE) is the most reliable tools. But we were unable to use it for this thesis work although we planned to use it. As an alternative, we used MUSE compatible version of AI based model to extract ECGs parameters. The analysis of ground truth information (ECGs parameters) extracted using AI based model was promising and useful to verify how correctly the ECGs are generated by the proposed model based on the given conditional information. The different evaluation methods that we used in this section were part of our third objective defined to achieve the main research question.

In our next chapter, we will discuss the results from a critical thinking perspective, for example, why the diffusion models' output is not good, etc. Similarly, we will also explain how this work could be extended in the future to overcome the weakness/limitation of the current work.

Chapter 5

Discussion and Future work

We have already presented our result in chapter 4 from different perspectives. In this section, we are providing critical analysis of our results. Here the term critical thinking refers to how good our result is overall, what is most impressive part, what the problems in existing models, why some models perform bad, what could be done improved etc. Similar, we will also explain about how we can extend this work in future even in more efficient way.

5.1 Discussion

After reading the result section, different questions may be raised in your mind, for example, why diffusion models are not efficient? This is just a representative example. We would like to analyze the different from critical thinking perspective in this section. To make you easy to follow, we have presented our discussion over different sub-sections.

5.1.1 Choice of Conditional Parameters

We first started our experiments with gender (male/female) information. Then, we use class labels associated with each ECG. We have used five classes (NORM, MI, STTC, CD, HYP). When we checked the result, we figured out that the model has generated ECG signal. But we are not sure about whether the generated signal belong so given class/gender or not. We could compare the distribution of generated data of different classes and how they differ. But we could not verify in a reliable way. We discussed the result with doctors, and they gave us feedback that instead of using those parameters which are difficult to verify, and most importantly lower use cases, they recommend us to use ground truth related to ECG (eg. PR-duration, QT-interval, P-peaks etc.). Those ground truths are extracted using a medical tool called MUSE. The greatest advantage we will get from this is that we can enforce the model to generate the ECG of desire ground truth. This is more useful from a clinical perspective to be used in the development of AI model. The use of ECG ground truth (ECG features) as conditional information is more prominent than use of class labels/genders. Last but not least, as we said earlier, we can verify this statistically how correctly it has been learned.

5.1.2 Why Are Similarity and Accuracy Metric not Improved in Conditional Models?

There might be several reasons behind the weaker/mixed result of conditional models. We would like to mention some of them here. The most possible one is the complexity. Adding the additional layer of condition to be learned by and combining it with Gaussian noise increases the complexity of the model. It is possible. For example, if we provide different ground truth for a similar distribution of random noise, then obviously it increases some level of complexity for the model. Another possible reason could be that the training is not good enough to map the noise with ground truth properly. The working mechanism of GAN model is that it learns to map the random noise to given inputs, during this process, model learns the patterns and corrections of random noise and generates different output. If the model is not generalized enough, it will have a direct impact on output.

However, to come to the conclusion that conditional models produce a lower quality ECG is unfair. The quality of generated samples is always important in any kind of model. But the question is what is the single measurement that can tell the quality of samples? Normally, models yielding smaller FID are considered better, however, this is not fully true. When we checked our experiment results, we noticed that samples with a smaller FID are classified by model with higher accuracy. In our experiment results, we can see that in both datasets, conditional diffusion models yield better FID and accuracy metrics, i.e., based on FID and accuracy metric, the conditional models are winners. In the case of GAN, accuracy of conditional models is higher in both datasets; however, FID scores are mixed. In private dataset, conditional models have smaller FID scores while in the public dataset it is oppositive. This indicates mixed results.

Similarly, on top of quality, the major intention of using the conditional model is to generate the sample of desired choice, that is, enforce the model to generate ECG of given ground truth. To verify whether the model can generate associated ECGs for a given ground truth, we performed MUSE feature analysis. The analysis shows that both GAN and diffusion models are able to generate ECG whose ground truth are similar to input ground truth. This provides the flexibility to users to generate ECGs of their requirements.

5.1.3 Design of Discriminator Architecture

We tried different approaches to pass the conditional information to the discriminator. The first approach we tried was similar to the generator architecture, that is, passing the ground truth information in each block. This did not work. Then, we tried to pass the ground truth as combined with input data as additional channels. To be added as a channel, it needs to have 5000 points. Therefore, we let the model learn. Unexpectedly, both methods did not work on GAN models. The possible reason could be complexity. This is the perfect example to show how adding one extra parameter completely collapses the GAN model. However, a similar approach works for diffusion models. The training loss of discriminator is presented in Figure 5.1 which shows how the models collapse. Then, we use a simple approach, i.e., pass the same information on the input as one additional channel by repeating ground truth. This method simply works.

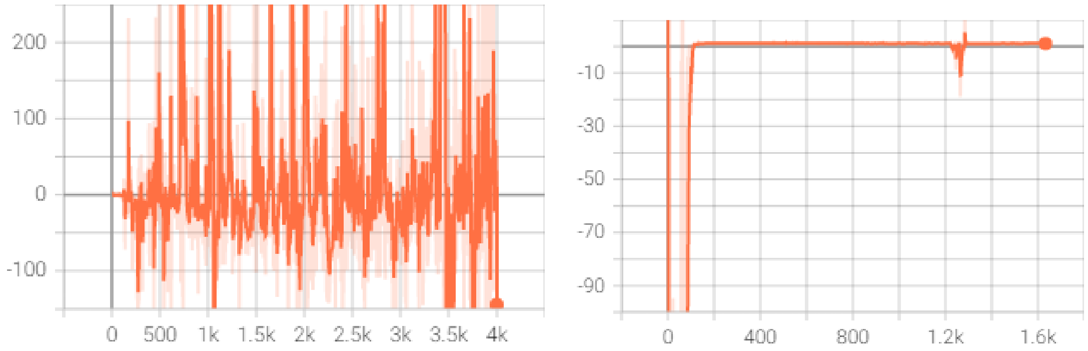


Figure 5.1: Discriminator failer when passing ground truth as learnable parameter. (left): passing ground truth at every block of discriminator, (right)-passing ground truth without normalization.

Similarly, we have tried different architecture of discriminators. The first architecture that we tried was a fully convolution-based discriminator. This kind of discriminator was used in different research works (e.h. DCGAN [73], StyleGAN [45]) related to image generation. Similarly, we have also tried the Patchgan style of discriminator which is used in image generation by Pix2PixGAN [115]. When we analyze the graphs generated at the validation step during training, we observe noticeable mistakes in generation, for example, issues in generating last part of data, r-peaks are generated just after the r-peaks with varied heights. Therefore, we decided to use similar architecture as used in pulse2pulse.

5.1.4 Why IS Score Is Lower in Private Dataset?

The information score is one of the good measures to check the quality and diversity of generated and fake samples. In the case of private datasets, compared to open source datasets, the value of IS is relatively small. This is the interesting information to be discussed. As we said earlier, it is calculated based on two things, i.e., quality, which is checked by lower entropy score (if each distribution belongs to one of the class labels) and diversity (the distribution of class labels, i.e., different variant). The quality somehow is measured by similarity (FID) score, if the output belongs to one of the classes, i.e., the output looks the same as the input, which results in small FID score. However, the only thing that is not verified is diversity. In this scenario, we could guess that the real data that we have used for training do not have many different patterns. When we discussed this finding with experts, they also verified that the open dataset (PTB-XL) also has different kinds of unusual patterns. In the ECG domain, those unusual patterns are called horse and zebra. One of the representing samples of the real ECG is presented in the figure below.

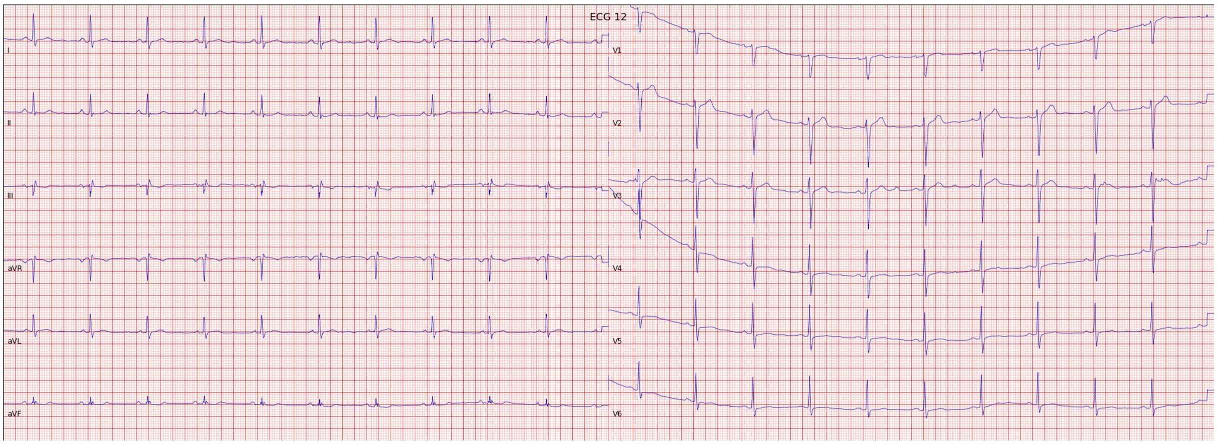


Figure 5.2: Representative sample of unusual pattern exits in open source dataset.

5.1.5 Precision Score vs Recall Score

If you observe the result of precision and recall score presented in table 4.4 and 4.5, we can see the different patterns in precision and recall score. More specifically, in the case of open source dataset (PTB-XL), ML classifier models produced relatively slower recall scores in both training settings except 12 for an exceptional result in normal training (model:*AdvP2P-SD*, PS:0.93, RS:0.96) and conditional training (model:*AdvP2P-AutoEmb*, PS:086, RS:0.87).

Having a lower recall score indicates that the model mostly misclassified positive samples (i.e., real ECG) as negative samples (i.e., fake ECG). There could be two possible reasons behind this result. The first one is that generated fake samples are good quality so that the model gets confused in classification. But the important thing to note is that if the recall score is lower because of fake quality, then the precision score also needs to be lower, that is, fake also needs to classify as real. But it seems that is not the condition, because the precision score is very high in some of the models (*AdvP2P-PosEmb* , *AdvP2P-NtF-AutoEmb* in the normal training setting). This leads to the second possible reason that the model cannot identify all real samples properly so that the ECG signals of unusual patterns (horse and zebra) are classified as fake.

In the case of precision and recall scores, having higher differences between these two scores is not a good sign. The ideal condition in our case is to get both scores relatively small. If you observe the conditional model's results on PTB-XL dataset, the precision and recall score are quite similar, which indicates that adding conditional information helps to balance the generation of different patterns of data.

On the other hand, the precision score is relatively smaller in private dataset which is just an oppositive condition as compared to open-source dataset. This indicates that most of the fakes are classified as reals, but not the same proportion of reals are classified as fakes. The possible reason is that the model has generated some data patterns perfectly, but not all of the different patterns that exist in the real dataset. This is exactly the case of the diversity issue in the GAN model. This argument is also supported by latent space/feature visualization as well. However, the interesting thing is that, compared to the open-source dataset, the model produces a relatively smaller precision and recall scores in the private dataset, indicating that a high proportion of real and fake are misclassified in the private dataset.

5.1.6 Can We Use Generated Synthetic ECG for Training AI Models?

After training the model, we generated the 10k samples from each model over different checkpoints. After randomly checking the 100 samples from 10k fake samples from each model, we figure out that even the best model has some problems (noisily signals) in generating the realistic ECG signals. This is exactly the reason we are not getting better results in the ML classifier. Therefore, instead of directly using the generated fake samples for training AI-models, we can extract the ground truth from fake samples and check if the ground truth falls in between the range of real ground truth. If the extracted ground truth looks normal, we can use those samples for training AI-models. This will help to mitigate the possible chances of misguiding the AI-models.

5.1.7 Diffusion vs GAN: Which One To Choose?

Although we got a relatively lower overall quality of the ECG in both training settings, we are very optimistic with the diffusion models. There are several reasons behind this, training the diffusion models are very easy as compared to GAN model. This can be in terms of time and complexity. We had an experience where GAN models take more than 10 days (about 1 and a half weeks) to complete the training, however, diffusion-based models were trained in 2/3 days. Similarly, the biggest issue we faced is the complexity or uncertainty issue in GAN. A very small change in model results in model collapse. For example, while training the conditional discriminator, the model collapsed. We tried different runs to be sure that it was not because of randomness, however, later we normalized the conditional information and it started to work. Similarly, when passing the conditional information to the discriminator, we first passed as a learning layer, i.e., take conditional information as input (5/7 parameters) and converted into 5000 parameters, and finally concatenate as one channel. This seems to be a more advance way than repeating the same information and making it 5000 points. These are just representative examples. Based on the result from both models, GAN models are clear winners, and we should use the data generated from GAN models for further use. However, there is an equal opportunity to further improve the GAN and research in the direction of the diffusion model for time series. By considering the simplicity of training, diversity in output, and not heavily dependent on large samples, researching in the direction of diffusion might be a good idea.

But you may point out that the results from diffusion models are of lower quality than GAN, in such a case why focus on diffusion-based models. We think this is very important to clarify. We have analyzed the possible reasons behind the result of diffusion results in the below section.

5.1.8 Why Diffusion Based Models Are Weak?

After running several experiments and checking different metrics, we got the impression that the results of GAN models are more promising than diffusion-based models. Based on this information, if we conclude that the diffusion model is not good, this will be unfair. Therefore, we would like to discuss why diffusion-based models are not as good as we were expecting. The first and most important one is the architecture; we have adopted the exact same architecture as used in the original paper, which was proposed for the image dataset. Maybe changes to the model architecture are necessary. Another important thing is that we have used default model parameters like scheduler (cosine), time steps of 500, and uniform noise of -1 to 1. These parameters could have a higher impact on the result. Let us share more details, for example, in the original paper Gaussian noise is used as input during generation. In the case of ECG, it did not work, so we changed to a uniform distribution between -1 and 1. It is essential to verify what the distribution of the ECG signal will be after adding noise so that we can peak the similar distribution of noise as well when generating new samples. We were unable to manage all these during the thesis work, but it could be interesting to check this in the future. We also tested time steps 500, 1000, and 2000. From the analysis, we found out that 500 is good. But we did not check less than 500, maybe 250/300 is enough in the case of ECG. The effect of these parameters should be checked in future work.

Another interesting and most important observation is that in the case of image generation, generating images of varied size at varied locations is quite OK as far as image is clear. However, these properties may not work in the case of ECG because ECG has a fixed pattern. Generating from random noise may not lead to generating ECGs of perfect pattern. Similarly, diffusion models do not consider conditional information while optimizing the loss, which leads to the situation that conditional information may not be learned by the diffusion model perfectly as in GAN.

5.1.9 What Could Be The Issue With Proposed Architecture?

There could be two main drawbacks to the proposed GAN architecture. The first is longer training time. Our proposed generator has a U-NET based architecture, which consists of an encoder part and a decoder part. Because of this, the model becomes heavy in terms of the number of learnable model parameters and results in longer training time. Another possible issue could be the difficulty in inference. In the current proposed architecture, the model takes 8 × 5000 random noise as input and generates an output of the same size. In such an architecture, it is very difficult to correlate the relation of input with output. For example, if you want to see the effect of some parameters, you can only change those parameters and re-generate fake samples. In such a case, we might get similar ECG with some changes (smaller/bigger p-waves, smaller/bigger QS-interval, etc.). However, it is difficult with the current architecture, as the model has huge parameters as input.

The solution to both issues discussed above can be solved using only decoder architecture. Here the decoder-based architecture means the generator model takes latent space (i.e., a small number of noise - 100/256 points) as input and enlarges to the required output shape. This is the most commonly used style in GAN. As the generation starts with a small number of noises, it is possible to figure out the relationship of noise with different features of the ECG so that by changing only those points we can generate the ECG of our choice.

5.1.10 Overall Impression

In the previous chapter, we evaluated the synthetic ECGs signals generated by proposed models in different ways. Similarly, we provide more discussion on the results in this chapter. The results and analysis show that overall the data generated by proposed models are promising and, most importantly, the models are generating ECGs according to the given conditional information. Experts' feedback of data generated by different models further justifies that the proposed models are good at generating data for the given condition.

However, the data generated by all models are not perfect. After the careful observation of generated fake ECGs, and feedback from experts, we are discussing some of the odd data as well so that you get a better picture of the weakness of model as well. Figure 5.3 is one of the example of noisy ECG with strange T-waves. Similarly, the R peak of $V1$ channel is completely odd.

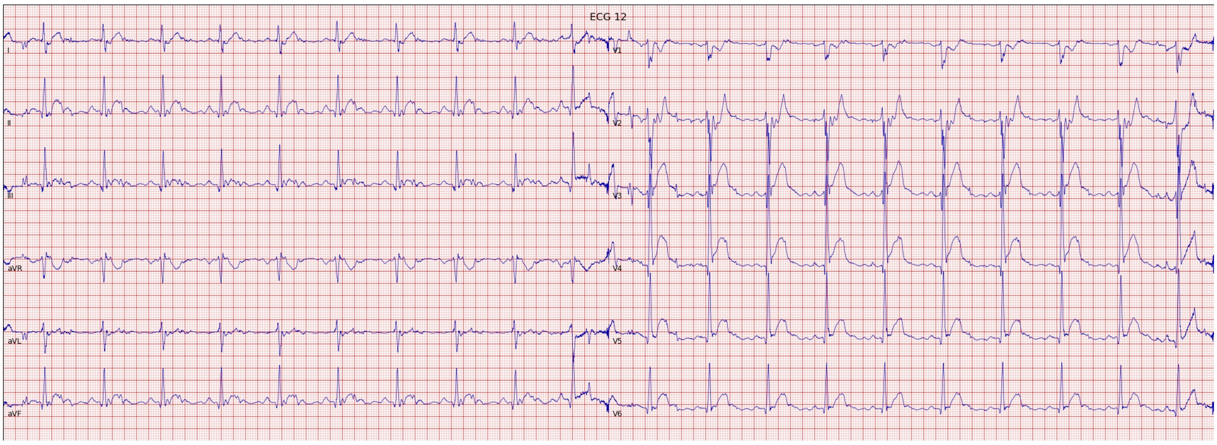


Figure 5.3: Example ECG of noisy signal, bad R peaks at V1 channel, strange T-waves.

Figure 5.4 is a perfect example of long pause and fast heart beats. In some places, the heartbeats are very frequent, while in some places there is a long pause. Although the waves are generated perfectly, heart beats pause and fast beat within small time interval is not possible in real scenarios.

5.1.11 Challenges, Pitfalls, and Lessons Learned

There are many challenges, pitfalls, and lessons learned experience while working on this thesis work. This is the first time I have worked in the field of generative models. I had no experience of working with GANs. Interestingly, I had not even heard of diffusion models before starting this master thesis. Therefore, in the start it was very challenging to make the model work. I realize working on the generative more is much more difficult than working with regression and classification problems. I tried out many different techniques in building model architecture, for example, using inception-style model architecture especially when expanding channels. It did work out. Adding one layer led to the model collapse condition. It is often difficult to figure out what is wrong with the model. Similarly, GPU memory issue is another common problem we face. Similarly, training time is another huge issue. The GAN model takes around 10 days to complete training. It is often difficult to say whether the model works or not by looking at the starting few epochs. Another important challenge is

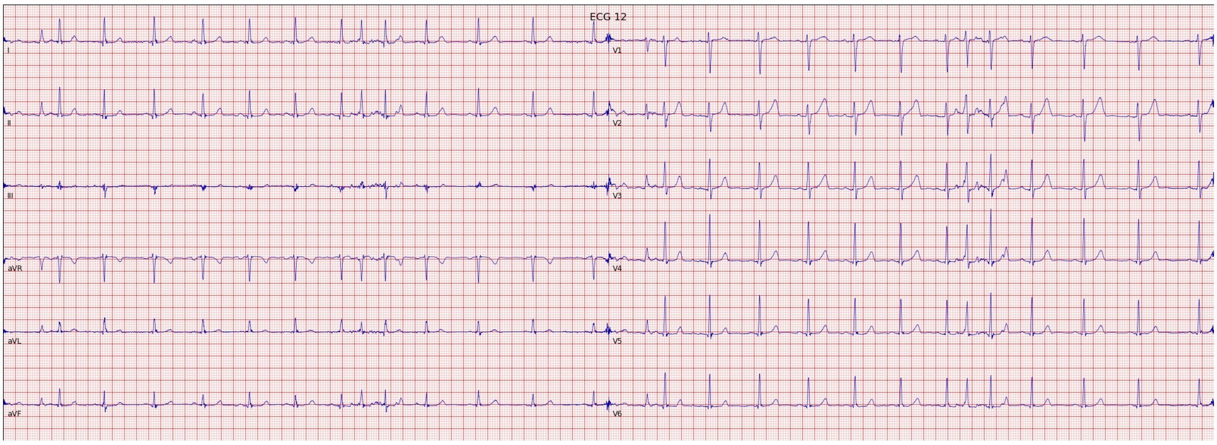


Figure 5.4: Example ECG of having long pause and unexpected fast heart beats.

to understand the ECG by heart. This is the first time I am working with ECG data. Therefore, I mostly face problems in validating time. If the generated ECG is random then I can understand that the model is not working but it is very difficult for me to validate specific waves, for example, P-waves, QRS duration, T-wave, QT-interval, etc.

This thesis work has helped me to build a strong foundation for working on generative models. I have learned many lessons that will be fruitful in my future career. I realized the great role of normalization in the generative model. The better normalization technique for input data and normalization in-between the model architecture play an important role. Similarly, it is often difficult to figure out where are problems. Therefore, it is very important to start with small model architecture and add layer one by one. Similarly, to mitigate the long training times, I should think about distributed training concept (training one multiple GPU) or designing the efficient model architecture.

5.1.12 Contribution to Research Community

This research work is one of the great efforts to mitigate the research gap in the direction of generating realistic synthetic ECGs signals guided by the given ground truth information. This research provides flexibility to healthcare professionals and researchers in generating fake ECG based on their defined ECGs parameters. The data generated by the proposed model can be used for different purposes. For example, it can be used to build an efficient method for validating synthetic data, training AI models, testing software, sharing the data for research purposes, and many more. However, the greatest benefit is the ability to generate data with ground truth. The is very beneficial for building AI-based diagnosis tool. If there is lack of particular kinds of data, we can use proposed model for generating that specific group of data so that it will the model to become more generalized and mitigate the issue of bias and overfitting.

5.2 Future Work

In this thesis work, we have worked with two different generative models under two different training settings, i.e., generative models with conditional information and without any conditional information. To compare and see the consistency of model performance on different datasets, we have trained our models on two different datasets. Similarly, on the evaluation sides as well, we are not limited to checking only the similarity distance. We used different approaches to evaluate the generated samples. We have done a lot of work on this thesis. However, we believe that we could extend this work in the future for improvements. We would like to present some interesting ideas to further improve the model.

5.2.1 Loss Function Modification

The first interesting and simplest idea that could help to improve the generator is modifying the loss function. In our current training setting, we have used the Wasserstein loss as shown in the following equations **??** and 5.2 to update the discriminator and generator.

$$L_D^{WGAN} = E[D(x)] - E[D(G(z)] + \lambda E[(| \bigtriangledown D(\alpha x - (1 - \alpha G(z)))| - 1)^2] \tag{5.1}$$

$$L_G^{WGAN} = E[D(G(z)] \tag{5.2}$$

We believe the generator loss could be improved by adding other loss functions in addition to WGAN generator loss. For example, the addition of mean square error (MSE) between the real and generated fake samples could help to improve the ECG signal at each point level so that the loss between different waves is correctly measured and improved. This kind of technique is used in image generation as well [102]. This is even more relevant in the case of ECG because ECG has a standard pattern. The addition of MSE may help to correctly generate the peaks of different waves. So, we can modify the previous generator loss as presented in eq. 5.3.

$$L_G^{WGAN} = E[D(G(z)] + \sum_{i=1}^{D}(x - G(z))^2 \tag{5.3}$$

5.2.2 Make Conditional Information Part of Loss Function

Similarly, if we want, the model learns the given ground truth perfectly so that it generates the signal of the same ground truth. We may need to consider the ground truth as a part of loss function. It is not possible that the MUSE tool is used during the training phase to calculate the ground truth. Therefore, we can come off with a new idea using ML model for predicting ground truth and compare it with original ground truth. This might help to generate the ECG of required ground truth. In addition to passing the ground truth information with given input, calculating loss on the given ground truth and ground truth of generated fake signal helps to better generate the fake sample of the required ground truth. This idea will help address the most common issue of strange P-wave, T-wave, unexpected pause, etc. The above generator loss function can be modified as presented in eq. 5.4. In the equation the term gt refers to ground truth.

$$L_G^{WGAN} = E[D(G(z)] + \sum_{i=1}^{D}(gt(x) - gt(G(z)))^2 \tag{5.4}$$

5.2.3 Transformers Based Architecture

In addition to the changes in the loss function, we could add a transformer block [97] to further improve the generation of ECG signals. Transformer based architectures are widely used in the generation of images [111], text [97, 112], time series data [40] etc. The concept of transformer is a more advanced concept than TCN. TCN captures the relationship of each point with different points in the past but not with every other point, while the transformer captures the relationship of each point with every other point in the given data.

5.2.4 Data Augmentation

During the analysis, we observed that the model is quite good at replicating the ground truth that belongs to the majority of samples. For example, in the analysis presented at one-to-one section of MUSE feature analysis, we saw that the models generate very close ground truth from the input ground truth with heart rate of 66-72. In the dataset, most of the ground truth belongs to this group. On the other hand, the minority ground truth is not properly learned. To overcome this issue, we could try different solutions. The very easy way is to add more samples belonging to minority cases if more data are available. Another worthy solution to be tried out is the use of augmentation techniques only to minority samples. What kind of augmentation works best is the topic of research; however, trying to add random noise, shifting window, etc. are some of the common techniques in the time series domain.

5.3 Summary

In this chapter, we discuss the results from different perspectives. More especially, we analyze the result from a critical point of view. This discussion gives a broader understating of the model and its possible weakness. Similarly, we also share the challenges and the lesson learning story of this thesis and how this thesis works contributes to the field of scientific computing. On the other hand, to overcome the weakness of current work, we have also presented some future steps in the future work section. The main finding of this thesis work is concluded in objective wise in the next conclusion section.

Chapter 6

Conclusion

6.1 Summary

Electrocardiograms signals are used for identifying the healthiness of a person. Decoding the ECGs signals and predicting the healthiness of a person is challenging and demands expert human resources. Several reports have shown that the lack of expert human resources in the healthcare sector is one of the major issues worldwide. To assist healthcare professionals and speed up the diagnosis process, the use of artificial intelligence has been increasing significantly. However, artificial intelligence-based models are highly dependent on data. The lack of data has become one of the major bottlenecks for building artificial intelligence-based tools. The use of synthetic data is emerging as an alternative solution. In the domain of ECGs, researchers have proposed different methods for synthesizing ECGs data. However, to the best of our knowledge, none of these methods consider generating ECGs based on conditional information such as desired heart rate, QRS duration, etc. Some researchers consider class labels as conditional information [2, 104]. However, controlling ECG generation based on ECG parameters is more advanced and more useful than class labels because after all class labels differences will be visible in ECG parameters such as heart rate. Therefore, generating synthetic ECGs signals based on a desired ECG parameter is even more beneficial.

We carry out extensive research and propose generative models based on generative adversarial networks and denoising diffusion models. We tested our proposed models with and without conditional settings on two different datasets. The experimental results show that the proposed models can generate very realistic ECGs signal which is even difficult for human experts to distinguish between real and fake. Similarly, a machine learning model trained on real and fake data shows that the model has a 21% error in distinguishing between real and fake, where the precision score of 0.69 and the recall score of 0.78 further prove that both real and fake are misclassified. Similarly, the analysis of a given input ECGs parameters as conditional input and ECGs parameters extracted from the associated synthetic ECGs proved that models are generating ECGs according to the given conditions. We have also validated the quality of data using FID distance, power spectrum analysis, expert review, and laten space visualization. All these evaluation methods confirmed that the quality of synthetic ECGs is similar to that of real ECGs. Interesting, out of the two generative models, GANs based models are more accurate in generating conditional information. Our proposed model can be used for generating data according to desired conditions and those data can be used for training AI models. Last but not least, these augmented data can be shared with other researchers without any concern of privacy.

The results of the proposed models are promising; however, the results can be improved further. We have presented different ways to improve this work in the further work section 5.2.

6.2 Revisiting the Problem Statement

The main research question what we have defined in this master thesis is *"**Can deep generative models generate realistic ECG signals based on the given condition information? **"*. To achieve the goal of research question, we defined three main objectives. The objective and the results associated with that objective are explained in the following paragraphs.

Our first objective is ***"Research and develop state-of-the-art deep learning based generative models (GAN and Diffusion) to generate realistic fake ECGs"***. To achieve this objective, we carried out extensive research on two state-of-the-art models called generative adversarial networks (GAN) and denosing diffusion. We proposed a UNET based GAN model architecture. We described out contribution related to each proposed model in the contribution section 6.3. The experimental result shows that FID scores of ECGs signal generated by proposed model are smaller in both datasets. Indeed, in PTB-XL dataset, the FID score of *AdvP2p* mode is 4.87 which is almost half than the normal TCN model (8.11). This demonstrates that we improved the model by adding a bidirectional concept. Similarly, we further improved the model by using the concept of embedding and feature learning (called Noise to Feature (NtF) converter) layer before passing it to the model architecture. In the private dataset, the ML model yields an accuracy of 89% for the data generated by the *AdvP2P* model, while the data generated by *AdvP2P-AutoEmb* and *AdvP2P-NtF-AutoEmb* models are classified with 0.75%. This similar kind of result exists in an open dataset. This indicates that adding embedding and NtF layers further improved the GANs model.

On the other hand, we adopted the denoising diffusion model used for image generation and proposed the modified version of DDM for time series. The experimental result shows that the diffusion model can generate an ECGs signal. However, the FID distance and different classification metrics show that ECGs signal generated by diffusion has the highest FID and accuracy scores compared to the GANs model. The power spectrum analysis shows that diffusion model has mostly problems in generating correct R peaks, P-waves, and W-waves. Based on the results of different evaluation methods, GANs based model architecture are better in generating ECGs signals in a normal training setting (i.e., without any condition) compared to the diffusion model.

Similarly, our second objective is ***"Research and develop a suitable method to add a conditional parameter on proposed generative model so that model can generate ECGs of given condition"***. To achieve this objective, we modified the proposed nonconditional model and added the conditional input parameters in the model. The experimental results based on FID and accuracy scores show that the diffusion model generates better ECG after adding the conditional information compared to non-conditional diffusion model. However, this is just oppositive in the case of GANs, i.e., adding conditional parameters adds complexity to the model. Based on similarity and accuracy scores, the conditional GANs models are not improved. On the other hand, although FID and accuracy are not improved in conditional GAN models, the main intention is to see whether the model can generate ECGs similar to the given condition or not. To verify this, we performed the MUSE feature analysis (ECG parameters analysis) in section, which demonstrates that the conditional models generate fake ECGs similar to the given conditional information. Indeed, GAN based models are better at generating fake ECGs of given conditions. Interestingly, experts also verified that the ECGs signal generated by conditional GAN model (*AdvP2P-NtF-AutoEmb*) is the best output.

Our third objective is ***"Research and develop different types of analysis methods for evaluating synthetic data"***. Validating synthetic data is the most challenging task. To achieve this objective, we have used different types of evaluation methods. The similarity distance calculated by FID and IS shown that proposed embedding based GANs model (*AdvP2P-AutoEmb, AdvP2P-NtF-AutoEmb*) yields smaller scores in both conditional and non-conditional models. We used the ml model to distinguish real and fake samples. The accuracy, precision, and recall scores shown that as usual embedding based GANs model yields better scores in both conditional and non-conditional setting. These evaluation methods show us overall how good are generated fake samples. But it cannot tell us which signals are corrected generated and where are problems. Therefore, we used power spectrum analysis, and ECG life cycle analysis, which shows that the proposed GANs model signals are very close to real signals in terms of generating different waves, but ECGs generated by the diffusion model have low R peaks and mostly problems in p and w waves. It is difficult to visually verify whether the conditional ECGs are similar to a given condition for the layman and even for experts. Therefore, we used an AI based model to extract ECG parameters and compared them with the given input condition. The result shows that GANs based models are better at generating the ECGs for a given condition. We tested the conditional results with three different scenarios. The given ECG parameters and the ECG parameters extracted from fake samples are very close in each case. However, the result is not perfect for every parameter, which can be improved in future work.

All the experimental result discussed based on three different objectives confirmed that we have achieved all objectives. Hence, different experiments results of our proposed solution demonstrates that the main research question of this work addressed, i.e., our proposed novel tool can be used to generate synthetic ECGs based on a given input condition.

6.3 Main Contributions

The main significant contribution of this thesis work is to research and propose state-of-the-art conditional deep generative models for generating ECGs signals based on a given ground truth information. To the best of our knowledge, this is a novel work which considers the ground truth as a conditional input. This work provides the flexibility to users to generate ECGs signals based on their desired ECGs parameters, such as heart rate. This is a more powerful technique than using class labels, gender, age groups, etc. as conditional information. Our proposed single work is good enough to generate ECGs belonging to other different groups, i.e., no need to build models based on different conditions. Most importantly, using such an approach is more reliable, and the output can be verified easily. Generating data is heavily dependent on model types as well. To figure out what kind of model is more suitable for generating ECGs signals, we extensively work with two state-of-the-art methods (diffusion and GANs). When we started working on this work, there was not a single work on the conditional denosing diffusion model in the ECG domain. We converted the diffusion base model into conditional diffusion. However, recently [2] has used a conditional diffusion model in ECGs, but their diffusion methods are different from the denosining approach.

Similarly, in proposed GANs models, we used the concept of bidirectional TCN which can learn to generate the next point by considering long-term decencies with other points from both directions. Similarly, we used the concept of two types of embedding called sinusoidal position embedding and self-embedding learned by the model which transforms the input noise to waves before passing it to the model architecture. The result of the experiment shows that proposed model yields better results than the normal TCN model without any embedding.

Another significant contribution of this work is the use of different evaluation methods. It is essential to correctly verify the data generated by the model so that it can be used further. We show that different methods help to better understand the fake ECGs from different perspectives. In addition to this, our contribution to the research community has already been discussed in section 5.1.12.

The main takeaway message from this thesis work can be summarized as:

- We can use the conditional generative models to generate ECGs signals based on a desired ECGs parameters.

- Generative adversarial networks (GANs)-based models are better in generating ECGs signals.
- Bidirectional TCN approach is better than normal TCN in generating ECGs signals.
- The use of embedding layers helped model in better ECG generation.
- Further research is necessary to improve this work.

Bibliography

[1] Edmond Adib et al. 'Synthetic ECG Signal Generation using Probabilistic Diffusion Models'. In: *arXiv preprint arXiv:2303.02475* (2023).

[2] Juan Miguel Lopez Alcaraz and Nils Strodthoff. 'Diffusion-based Conditional ECG Generation with Structured State Space Models'. In: *arXiv preprint arXiv:2301.08227* (2023).

[3] Santhosh Alladi. 'Augmenting Electrocardiogram Datasets using Generative Adversarial Networks'. PhD thesis. University of Minnesota, 2020.

[4] analyticsvidhya. 'Machine Learning vs Deep Learning vs Artificial Intelligence | Know in-depth Difference'. In: *analyticsvidhya.com* (2021). URL: https://www.analyticsvidhya.com/blog/2021/06/machine-learning-vs-artificial-intelligence-vs-deep-learning/.

[5] Karol Antczak. 'A generative adversarial approach to ecg synthesis and denoising'. In: *arXiv preprint arXiv:2009.02700* (2020).

[6] Martin Arjovsky, Soumith Chintala and Léon Bottou. 'Wasserstein generative adversarial networks'. In: *International conference on machine learning*. PMLR. 2017, pp. 214–223.

[7] Anmol Arora. 'Conceptualising artificial intelligence as a digital healthcare innovation: an introductory review'. In: *Medical Devices (Auckland, NZ)* 13 (2020), p. 223.

[8] Pragati Baheti. *Supervised and Unsupervised Learning [Differences & Examples]*. 2022. URL: https://www.v7labs.com/blog/supervised-vs-unsupervised-learning.

[9] Swati Banerjee and Madhuchhanda Mitra. 'Application of cross wavelet transform for ECG pattern analysis and classification'. In: *IEEE transactions on instrumentation and measurement* 63.2 (2013), pp. 326–333.

[10] Dor Bank, Noam Koenigstein and Raja Giryes. 'Autoencoders'. In: *arXiv preprint arXiv:2003.05991* (2020).

[11] David Bau et al. 'Seeing what a gan cannot generate'. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 4502–4511.

[12] Muriel Boulakia et al. 'Mathematical modeling of electrocardiograms: a numerical study'. In: *Annals of biomedical engineering* 38.3 (2010), pp. 1071–1097.

[13] Andrew Brock, Jeff Donahue and Karen Simonyan. 'Large scale GAN training for high fidelity natural image synthesis'. In: *arXiv preprint arXiv:1809.11096* (2018).

[14] Eoin Brophy. 'Synthesis of dependent multichannel ECG using generative adversarial networks'. In: *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2020, pp. 3229–3232.

[15] Eoin Brophy et al. 'Generative adversarial networks in time series: A survey and taxonomy'. In: *arXiv preprint arXiv:2107.11098* (2021).

[16] Giorgia Carra et al. 'Data-driven ICU management: Using Big Data and algorithms to improve outcomes'. In: *Journal of Critical Care* 60 (2020), pp. 300–304.

[17] Dingfan Chen et al. 'Gan-leaks: A taxonomy of membership inference attacks against generative models'. In: *Proceedings of the 2020 ACM SIGSAC conference on computer and communications security*. 2020, pp. 343–362.

[18] Xi Chen et al. 'Infogan: Interpretable representation learning by information maximizing generative adversarial nets'. In: *Advances in neural information processing systems* 29 (2016).

[19] GD Clifford and MC Villarroel. 'Model-based determination of QT intervals'. In: *2006 Computers in Cardiology*. IEEE. 2006, pp. 357–360.

[20] Thomas Davenport and Ravi Kalakota. 'The potential for artificial intelligence in healthcare'. In: *Future healthcare journal* 6.2 (2019), p. 94.

[21] J De Bie et al. 'Performance of seven ECG interpretation programs in identifying arrhythmia and acute cardiovascular syndrome'. In: *Journal of Electrocardiology* 58 (2020), pp. 143–149.

[22] Anne Marie Delaney, Eoin Brophy and Tomas E Ward. 'Synthesis of realistic ecg using generative adversarial networks'. In: *arXiv preprint arXiv:1909.09150* (2019).

[23] Peter J. Denning et al. 'Computing as a discipline'. In: *Computer* 22.2 (1989), pp. 63–70.

[24] Prafulla Dhariwal and Alexander Nichol. 'Diffusion models beat gans on image synthesis'. In: *Advances in Neural Information Processing Systems* 34 (2021).

[25] Chris Donahue, Julian McAuley and Miller Puckette. 'Adversarial audio synthesis'. In: *arXiv preprint arXiv:1802.04208* (2018).

[26] Ricard Durall et al. 'Combating mode collapse in gan training: An empirical analysis using hessian eigenvalues'. In: *arXiv preprint arXiv:2012.09673* (2020).

[27] Andre Esteva et al. 'Dermatologist-level classification of skin cancer with deep neural networks'. In: *nature* 542.7639 (2017), pp. 115–118.

[28] David Foster. *Generative deep learning: teaching machines to paint, write, compose, and play*. O'Reilly Media, 2019.

[29] Shuqing Gao et al. 'Public perception of artificial intelligence in medical care: content analysis of social media'. In: *Journal of Medical Internet Research* 22.7 (2020), e16649.

[30] Tomer Golany, Daniel Freedman and Kira Radinsky. 'ECG ODE-GAN: Learning Ordinary Differential Equations of ECG Dynamics via Generative Adversarial Learning'. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 1. 2021, pp. 134–141.

[31] Tomer Golany and Kira Radinsky. 'Pgans: Personalized generative adversarial networks for ecg synthesis to improve patient-specific deep ecg classification'. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 01. 2019, pp. 557–564.

[32] Tomer Golany et al. 'Improving ECG classification using generative adversarial networks'. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 08. 2020, pp. 13280–13285.

[33] Ian Goodfellow et al. 'Generative adversarial nets'. In: *Advances in neural information processing systems* 27 (2014).

[34] Google. 'GAN Training'. In: *https://developers.google.com/* (2022). URL: https://developers.google.com/machine-learning/gan/training.

[35] Shinichi Goto and Shinya Goto. 'Application of Neural Networks to 12-Lead Electrocardiography—Current Status and Future Directions—'. In: *Circulation Reports* (2019), CR–19.

[36] Ishaan Gulrajani et al. 'Improved training of wasserstein gans'. In: *Advances in neural information processing systems* 30 (2017).

[37] Kazi Nazmul Haque et al. 'Guided generative adversarial neural network for representation learning and high fidelity audio generation using fewer labelled audio data'. In: *arXiv preprint arXiv:2003.02836* (2020).

[38] Steven A Hicks et al. 'Explaining deep neural networks for knowledge discovery in electrocardiogram analysis'. In: *Scientific reports* 11.1 (2021), p. 10949.

[39] Jonathan Ho, Ajay Jain and Pieter Abbeel. 'Denoising diffusion probabilistic models'. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 6840–6851.

[40] Shenda Hong et al. 'MINA: multilevel knowledge-guided attention for modeling electrocardiography signals'. In: *arXiv preprint arXiv:1905.11333* (2019).

[41] Khondker Fariha Hossain et al. 'ECG-Adv-GAN: Detecting ECG Adversarial Examples with Conditional Generative Adversarial Networks'. In: *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*. IEEE. 2021, pp. 50–56.

[42] Huaibo Huang et al. 'Introvae: Introspective variational autoencoders for photographic image synthesis'. In: *Advances in neural information processing systems* 31 (2018).

[43] Phillip Isola et al. 'Image-to-image translation with conditional adversarial networks'. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1125–1134.

[44] Brian Kalis, Matt Collier and Richard Fu. '10 Promising AI Applications in Health Care'. In: *hbr.org* (2018). URL: https://hbr.org/2018/05/10-promising-ai-applications-in-health-care.

[45] Tero Karras et al. 'Analyzing and Improving the Image Quality of StyleGAN'. In: *Proc. CVPR*. 2020.

[46] Tero Karras et al. 'Progressive growing of gans for improved quality, stability, and variation'. In: *arXiv preprint arXiv:1710.10196* (2017).

[47] Wei-Yin Ko et al. 'Detection of hypertrophic cardiomyopathy using a convolutional neural network-enabled electrocardiogram'. In: *Journal of the American College of Cardiology* 75.7 (2020), pp. 722–733.

[48] VV Kuznetsov, VA Moskalenko and N Yu Zolotykh. 'Electrocardiogram generation and feature extraction using a variational autoencoder'. In: *arXiv preprint arXiv:2002.00254* (2020).

[49] VV Kuznetsov et al. 'Interpretable Feature Generation in ECG Using a Variational Autoencoder'. In: *Frontiers in genetics* 12 (2021).

[50] Joon-Myoung Kwon et al. 'Comparing the performance of artificial intelligence and conventional diagnosis criteria for detecting left ventricular hypertrophy using electrocardiography'. In: *EP Europace* 22.3 (2020), pp. 412–419.

[51] Lumen Learning. *Physiology of the Heart*. 2022. URL: https://courses.lumenlearning.com/boundless-ap/chapter/physiology-of-the-heart/.

[52] N Lebanon and N Hanover. 'How many deaths are due to medical error? Getting the number right'. In: *Eff Clin Pract* 6 (2000), pp. 277–283.

[53] Christian Ledig et al. 'Photo-realistic single image super-resolution using a generative adversarial network'. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 4681–4690.

[54] Rui Li et al. 'Linear attention mechanism: An efficient attention for semantic segmentation'. In: *arXiv preprint arXiv:2007.14902* (2020).

[55] Xiaomin Li et al. 'TTS-GAN: A Transformer-based Time-Series Generative Adversarial Network'. In: *arXiv preprint arXiv:2202.02691* (2022).

[56] Kevin J Liang et al. 'Generative adversarial network training is a continual learning problem'. In: *arXiv preprint arXiv:1811.11083* (2018).

[57] Steven Y Lin, Megan R Mahoney and Christine A Sinsky. 'Ten ways artificial intelligence will transform primary care'. In: *Journal of general internal medicine* 34.8 (2019), pp. 1626–1630.

[58] Jenny X Liu et al. 'Global health workforce labor market projections for 2030'. In: *Human resources for health* 15.1 (2017), pp. 1–12.

[59] Kanglin Liu and Guoping Qiu. 'Lipschitz constrained GANs via boundedness and continuity'. In: *Neural Computing and Applications* 32.24 (2020), pp. 18271–18283.

[60] Anega Maheshwari, Priyanka Mitra and Bhavna Sharma. 'Autoencoder: Issues, Challenges and Future Prospect'. In: *Recent Innovations in Mechanical Engineering*. Springer, 2022, pp. 257–266.

[61] Manitoba. 'Shortage of cardiac testing staff puts patients at risk, Manitoba union says'. In: *CBC News* (2022). URL: https://www.cbc.ca/news/canada/manitoba/shortage-cardiac-testing-staff-1.6457979.

[62] John McCarthy et al. 'Artificial intelligence (AI) coined at dartmouth'. In: *Retrieved October* 28 (1956), p. 2021.

[63] Scott Mayer McKinney et al. 'International evaluation of an AI system for breast cancer screening'. In: *Nature* 577.7788 (2020), pp. 89–94.

[64] Lars Mescheder, Andreas Geiger and Sebastian Nowozin. 'Which training methods for GANs do actually converge?' In: *International conference on machine learning*. PMLR. 2018, pp. 3481–3490.

[65] Bertalan Meskó, Gergely Hetényi and Zsuzsanna Győrffy. 'Will artificial intelligence solve the human resource crisis in healthcare?' In: *BMC health services research* 18.1 (2018), pp. 1–4.

[66] Mehdi Mirza and Simon Osindero. 'Conditional generative adversarial nets'. In: *arXiv preprint arXiv:1411.1784* (2014).

[67] Seyed Sajad Mousavi, Michael Schukat and Enda Howley. 'Deep reinforcement learning: an overview'. In: *Proceedings of SAI Intelligent Systems Conference*. Springer. 2016, pp. 426–440.

[68] John Paul Mueller and Luca Massaron. *Deep Learning for dummies*. John Wiley & Sons, 2019.

[69] Andres Munoz et al. 'Temporal Shift GAN for Large Scale Video Generation'. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2021, pp. 3179–3188.

[70] Packt. 'Understanding the limitations of autoencoders?' In: *packtpub.com* (2022). URL: https://subscription.packtpub.com/book/data/9781789536089/11/ch11lvl1sec51/understanding-the-limitations-of-autoencoders.

[71] Esteban Piacentino, Alvaro Guarner and Cecilio Angulo. 'Generating synthetic ecgs using gans for anonymizing healthcare data'. In: *Electronics* 10.4 (2021), p. 389.

[72] Silvia Pokrivčáková. 'Preparing teachers for the application of AI-powered technologies in foreign language education'. In: *Journal of Language and Cultural Education* (2019).

[73] Alec Radford, Luke Metz and Soumith Chintala. 'Unsupervised representation learning with deep convolutional generative adversarial networks'. In: *arXiv preprint arXiv:1511.06434* (2015).

[74] Aditya Ramesh et al. 'Hierarchical text-conditional image generation with clip latents'. In: *arXiv preprint arXiv:2204.06125* (2022).

[75] Keerthi G Reddy, P Vijaya and S Suhasini. 'ECG Signal Characterization and Correlation To Heart Abnormalities'. In: *International Research Journal of Engineering and Technology (IRJET)* 4.5 (2017).

[76] Robin Rombach et al. 'High-resolution image synthesis with latent diffusion models'. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 10684–10695.

[77] Olaf Ronneberger, Philipp Fischer and Thomas Brox. 'U-net: Convolutional networks for biomedical image segmentation'. In: *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. Springer. 2015, pp. 234–241.

[78] Angela M dos Santos, Sergio R Lopes and RL Ricardo L Viana. 'Rhythm synchronization and chaotic modulation of coupled Van der Pol oscillators in a model for the heartbeat'. In: *Physica A: Statistical Mechanics and its Applications* 338.3-4 (2004), pp. 335–355.

[79] Divya Saxena and Jiannong Cao. 'Generative adversarial networks (GANs) challenges, solutions, and future directions'. In: *ACM Computing Surveys (CSUR)* 54.3 (2021), pp. 1–42.

[80] Jürg Schläpfer and Hein J Wellens. 'Computer-interpreted electrocardiograms: benefits and limitations'. In: *Journal of the American College of Cardiology* 70.9 (2017), pp. 1183–1192.

[81] Mohammed Yousef Shaheen. 'Applications of Artificial Intelligence (AI) in healthcare: A review'. In: *ScienceOpen Preprints* (2021).

[82] Abdelrahman M Shaker et al. 'Generalization of convolutional neural networks for ECG classification using generative adversarial networks'. In: *IEEE Access* 8 (2020), pp. 35592–35605.

[83] Amanpreet Singh, Narina Thakur and Aakanksha Sharma. 'A review of supervised machine learning algorithms'. In: *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*. Ieee. 2016, pp. 1310–1315.

[84] Pratik Singh and Gayadhar Pradhan. 'A new ECG denoising framework using generative adversarial network'. In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 18.2 (2020), pp. 759–764.

[85] Satinder Singh, Andy Okun and Andrew Jackson. 'Learning to play Go from scratch'. In: *Nature* 550.7676 (2017), pp. 336–337.

[86] Konstantinos C Siontis et al. 'Artificial intelligence-enhanced electrocardiography in cardiovascular disease management'. In: *Nature Reviews Cardiology* 18.7 (2021), pp. 465–478.

[87] Stephen W Smith et al. 'A deep neural network learning algorithm outperforms a conventional algorithm for emergency department electrocardiogram interpretation'. In: *Journal of electrocardiology* 52 (2019), pp. 88–95.

[88] Joakim Sundnes, Glenn Terje Lines and Aslak Tveito. 'Efficient solution of ordinary differential equations modeling electrical activity in cardiac cells'. In: *Mathematical biosciences* 172.2 (2001), pp. 55–72.

[89] Christian Szegedy et al. 'Going deeper with convolutions'. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1–9.

[90] Tahmida Tabassum and Mohiuddin Ahmed. 'A Simplified Cardiac Conduction Model and Twelve-Lead ECG Generation'. In: *2020 International Conference on Computer, Electrical & Communication Engineering (ICCECE)*. IEEE. 2020, pp. 1–5.

[91] Luke Taylor and Geoff Nitschke. 'Improving deep learning with generic data augmentation'. In: *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE. 2018, pp. 1542–1547.

[92] Vajira Thambawita et al. 'DeepSynthBody: the beginning of the end for data deficiency in medicine'. In: *2021 International Conference on Applied Artificial Intelligence (ICAPAI)*. IEEE. 2021, pp. 1–8.

[93] Vajira Lasantha Thambawita et al. 'DeepFake electrocardiograms: the key for open science for artificial intelligence in medicine'. In: *medRxiv* (2021).

[94] Ngoc-Trung Tran, Tuan-Anh Bui and Ngai-Man Cheung. 'Dist-gan: An improved gan using distance constraints'. In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 370–385.

[95] Alan M Turing. 'Computing machinery and intelligence'. In: *Parsing the turing test*. Springer, 2009, pp. 23–65.

[96] Antti Väänänen et al. 'AI in healthcare: A narrative review'. In: *F1000Research* 10.6 (2021), p. 6.

[97] Ashish Vaswani et al. 'Attention is all you need'. In: *Advances in neural information processing systems* 30 (2017).

[98] Brian Wahl et al. 'Artificial intelligence (AI) and global health: how can AI contribute to health in resource-poor settings?' In: *BMJ global health* 3.4 (2018), e000798.

[99] Yaohui Wang et al. 'Imaginator: Conditional spatio-temporal gan for video generation'. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2020, pp. 1160–1169.

[100] Lilian Weng. 'What are diffusion models?' In: *lilianweng.github.io* (2021). URL: https://lilianweng.github.io/posts/2021-07-11-diffusion-models/.

[101] Maciej Wiatrak, Stefano V Albrecht and Andrew Nystrom. 'Stabilizing generative adversarial networks: A survey'. In: *arXiv preprint arXiv:1910.00927* (2019).

[102] Junfeng Wu et al. 'ESGAN for generating high quality enhanced samples'. In: *Multimedia Systems* 28.5 (2022), pp. 1809–1822.

[103] Naren Wulan et al. 'Generating electrocardiogram signals by deep learning'. In: *Neurocomputing* 404 (2020), pp. 122–136.

[104] Yong Xia, Wenyi Wang and Kuanquan Wang. 'ECG signal generation based on conditional generative models'. In: *Biomedical Signal Processing and Control* 82 (2023), p. 104587.

[105] Li-Chia Yang, Szu-Yu Chou and Yi-Hsuan Yang. 'MidiNet: A convolutional generative adversarial network for symbolic-domain music generation'. In: *arXiv preprint arXiv:1703.10847* (2017).

[106] Fei Ye et al. 'ECG generation with sequence generative adversarial nets optimized by policy gradient'. In: *IEEE Access* 7 (2019), pp. 159369–159378.

[107] Hiroshi Yoshida et al. 'Automated histological classification of whole-slide images of gastric biopsy specimens'. In: *Gastric cancer* 21.2 (2018), pp. 249–257.

[108] Vasily Zadorozhnyy, Qiang Cheng and Qiang Ye. 'Adaptive weighted discriminator for training generative adversarial networks'. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 4781–4790.

[109] Joseph P Zbilut, Michail Zak and Ronald E Meyers. ‘A terminal dynamics model of the heartbeat’. In: *Biological cybernetics* 75.3 (1996), pp. 277–280.

[110] Junhai Zhai et al. ‘Autoencoder and its various variants’. In: *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE. 2018, pp. 415–419.

[111] Han Zhang et al. ‘Self-attention generative adversarial networks’. In: *International conference on machine learning*. PMLR. 2019, pp. 7354–7363.

[112] Hanqing Zhang et al. ‘A survey of controllable text generation using transformer-based pre-trained language models’. In: *arXiv preprint arXiv:2201.05337* (2022).

[113] Shengjia Zhao, Jiaming Song and Stefano Ermon. ‘Towards deeper understanding of variational autoencoding models’. In: *arXiv preprint arXiv:1702.08658* (2017).

[114] Fei Zhu et al. ‘Electrocardiogram generation with a bidirectional LSTM-CNN generative adversarial network’. In: *Scientific reports* 9.1 (2019), pp. 1–11.

[115] Jun-Yan Zhu et al. ‘Unpaired image-to-image translation using cycle-consistent adversarial networks’. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2223–2232.