# An evaluation of using transformer networks for ECG Analysis

Syeda Ambreen Yawar


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

Department of Computer Science
Faculty of Technology, Art and Design

OSLO METROPOLITAN UNIVERSITY

Spring 2023

# An evaluation of using transformer networks for ECG Analysis

Syeda Ambreen Yawar

An evaluation of using transformer networks for ECG Analysis

http://www.oslomet.no/

Printed: Oslo Metropolitan University

# Abstract

Electrocardiogram (ECG) is a simulated recording of heart activity in electrical signals. It carries essential clinical information in the form of amplitude and timing. It is used to monitor and analyze the functionality of the cardiovascular system by doctors in the health care department. The high potential for human error due to skills, knowledge, and workload stress in manual analyzing ECG can lead to morbidity and mortality in patients. Therefore, automatic aids are required that can overcome human errors, eases the load on doctors, and help doctors in the diagnosis of heart diseases. Artificial intelligence-integrated systems for ECG analysis are trained on millions of datasets and have seen more ECGs than doctors can see in their entire careers. Furthermore, it provides results in seconds, and no other factors like environment, workload stress, etc. affect the accuracy of the results. Therefore, AI-integrated ECG analysis are more accurate, quick, and reliable than doctors' manual analysis. Recently, deep learning-based tools have been the great attention as an aid to doctors towards accurate analyzing, annotation, and interpretation of ECG data. In deep learning models, Transformer Networks have become reference models with superior performance on different natural language processing and vision tasks. In this work, an evaluation of transformer networks is provided for the analysis of ECG. For this, two end-to-end deep learning frameworks are implemented for measuring the relevant intervals and amplitudes from the ECGs. The frameworks incorporated the transformer network and the multi-layer perceptron to attend to the information stored in ECG signals and predict the relevant (amplitudes and intervals) values. However, both frameworks use different variants of the structure of the transformer to each other.The first framework utilizes the encoder-only structure while the second utilizes the encoder-decoder structure of the transformer network. To overcome the problem of privacy issues (collecting and sharing among researchers) in health care, the implemented framework is evaluated over a realistic synthetic ECG dataset called DeepFake ECG. According to the training results, the performance of encoder-only beat the performance of encoder-decoder in ECG analysis. This deep learning model can aid doctors to perform heart disease diagnosis and improve the health care system's efficiency.

# Acknowledgments

I am extremely grateful to my Supervisor for this work, Pål Halvorsen (Professor, Chief Research Scientist/Research Professor) for his guidance during the project.

I am extremely grateful to my Assistant supervisor of this work, Vajira Thambawita (Research Scientist At Simula Research Laboratory) whose knowledge of the subject steered me through this research.

I would like to express my deepest appreciation to my Assistant supervisor of this work, Steven Hicks (Research Scientist At Simula Research Laboratory) for being instrumental in defining the path of the project.

Special thanks to Kløfta Biblioteket for providing me with the workspace for researching this work.

I want to acknowledge Professor Raju Shrestha (Professor, TKD, OsloMet ) for inspiring my interest in deep learning techniques.

A special thanks to my colleague (students at OsloMet) for their support throughout the journey.

Lastly, I would like to mention my beloved family, and my dear husband without his support this endeavor would not have been possible his understanding and patience for the many hours I spent sitting in front of the computer for researching and developing this work, my lovely kids who bear with me in this whole journey especially works on weekends and my siblings and parents for their nonstop moral and emotional support.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Cardiovascular diseases (CVDs) are one of the biggest causes of death in the world. According to World Health Organization (WHO), in 2019, an approximation of *17.9* million people, out of which *70%* were under the age of *70*, died because of different CVDs [47]. Cardiac means relating to the heart and vascular means related to blood vessels, thus, cardiovascular (CV) refers entire system of the heart, veins, and arteries while CVDs refers to the group of disorders of the heart and blood vessels.

Myocardial Infarction (MI) or heart attack, is also one of the biggest causes of death and disability of patients globally [41]. It happens for a very short interval of time to the patient but this small event of heart attack if gone undetected could cause death to the patient. The cause of heart attack is the presence of a long-duration of inflammation on the vascular wall. This inflammation refers to another CV which is called Ischemia. Ischemia is the imbalance between the supply and demand of oxygen and blood to the heart due to clogged arteries. If early detection of myocardial ischemia happens then there are significantly low chances of heart attack. Symptoms of Myocardial infarction usually last for a minimum *20* minutes and contain various discomfort on the chest, jaw, arm, etc. These symptoms are sometimes misdiagnosed with other diseases like gastrointestinal, muscular disorder, etc. or it sometimes happens to patients without any symptoms that made it more difficult to diagnose. The simplest diagnosis for myocardial infarction is the testing of heart activity with the help of an electrocardiogram (ECG) recording which provides the details of the history of past myocardial ischemia and, if hear attack happened in the near past or is soon to happen to the patient. Therefore, the early diagnosis can be done with the help of the ECG signal of the patient [41].

ECG is the recording of the electrical activity of the heart of the patient and is illustrated in the form of a signal. In figure 1.1 an illustration of the ECG signal with its annotation is provided. Cardiologists interpret the relevant amplitudes, frequencies, and duration of some waveforms for the diagnosis of CVDs. This

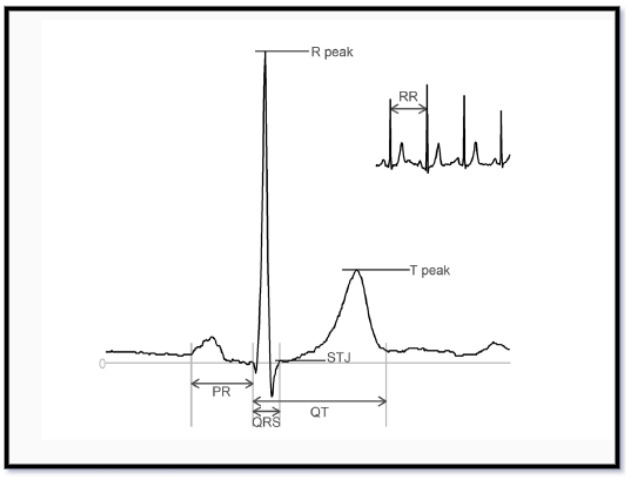


Figure 1.1: An annotated picture of ECG signal (taken from [15])

process of analyzing ECG signals for interpretation is called ECG analysis [3]. For example, if the ECG like 1.1 has the high peaked *T* waves (which is a relevant feature in ECG signal) before *ST-elevation* (which is again a relevant feature in ECG signal) then the interpretations gives the early diagnosis of acute heart attack [50]. The analysis of such relevant amplitudes, duration, and intervals from ECG is key for diagnosing CVDs. This task requires a tremendous amount of time and expertise from cardiologists to explore the ECG signal and find out the relative information about CVDs from ECG. Mostly, cardiologists end up with errors in reading features manually which results in failed diagnosis which is risky for patient health. From the above example of ECG analysis, we can say that how much the amplitude of the peak of *T* wave is considered a threat to heart attack and any calculation error might risk the patient's health. At this stage when there are chances of error, the computer science field helps the medical science field by providing aid to doctors with the research and developments of automatic analyzing systems for ECG analysis. The automatic analysis measures the relative information and provides it in digitized and in better interpretable ways. In addition, in recent times, AI researchers are attracted to aid cardiologists by improving the system by making it more efficient and intelligent. Several machine-learning modules and recently, majorly deep-learning modules have been researched and utilized for ECG analysis which is continuously improving the system efficiency.

The reason behind providing such information about cardiovascular diseases, its diagnosis using ECG analysis, with the possibility of error by the cardiologist while reading it, and the development of automatic and efficient tools for ECG analysis as an aid to the medical sector for its efficiency, is simply carving the ground

for understanding the motivation behind this thesis work which is provided in next section.

## 1.1 Motivation

ECG analysis done by clinicians manually is a traditional method and is widely used in the health sector. However, the results from such manual analysis are highly dependent on the clinician's skills, knowledge, and concentration in stressful work environments while analyzing, the chances are high for human error and inaccuracy in ECG analysis. As the accuracy in the analysis of ECG is directly related to the diagnosis of CVDs and patients' life, therefore, accurate analysis is required. In addition, manual ECG analysis consumes a big amount of valuable time for the clinician to investigate the long-length ECG graph manually and count the big number of small boxes for measurements which are complicated and requires effort from the clinician, and again chances for human error are high. Therefore, automatic ECG analysis is required to shed the workload of the clinicians and provide effortless and timely analysis of ECG. Furthermore, if this automatic analysis is integrated with AI algorithms that are trained on a big ECG dataset, then it will work as an assistant to clinicians and provide more accurate analysis where there will be no workload stress and time consumption involved from the clinicians. AI-aided ECG analysis methods are becoming better in performance and popular to use day by day by clinicians and it has attracted the research community to work on for better AI algorithm for automatic ECG analysis.

Several famous and well-performed deep-learning modules from different sub-field of artificial intelligence like convolution neural network (CNN) [27] from computer vision (CV), recurrent neural network (RNN) [37] from natural language processing (NLP), etc., are ton-wised researched for ECG analysis [17]. Furthermore, the mechanism of attention which improved the quality of dependencies of the model has attracted researcher to include it in above mentioned modules for better results. There is a network in NLP that solely depends upon its advanced attention called multi-headed-self-attention and parallel computation of sequence-to-sequence task has revolutionized the field of NLP and beat the RNN and its advanced modification with its superior performance, this network is called Transformer network [44]. Soon after the publication of the transformer network, it is widely been used in almost every field of AI. Though transformer network has given superior results, still, in ECG analysis it is comparatively been researched very less than the other modules of deep learning. Even an article for the collective survey about the transformer network and its modification for the ECG-related task are not been done by the researchers

yet. Therefore, the scope of the research for transformer networks for ECG analysis is quite big and requires a lot of effort from the researchers.

Hence, one of the motivations of this thesis work is to fill the gap in research and put efforts into researching transformer networks for ECG analysis. Since the transformer network has provided superior results in other tasks, thus, another motivation is to research the efficient, intelligent, AI-aided ECG analysis system that can aid cardiologists with accurate analysis of ECG and reduce the time-cost and risks of misdiagnosing patients. Finally, the overall motivation of this work is to help the health sector for increasing its efficiency in providing healthcare necessities to its users.

## 1.2 Problem Statement

In recent years, the transformer networks, known for its encoder-decoder structure with its powerful multi-head self-attention mechanism [44], have widely been used in almost every domain of computer science due to the superior performance shown by it. Several different modifications to the structure of transformer networks have been proposed by the researcher for the designated tasks [9]. For example, it has highly been researched for its variants in structures like encoder-decoder [26], encoder only [29], structure for the text generation task of NLP [7]. However, when looking into the research field of automatic analyzation ECG, the transformer networks have relatively been less researched, discussed, and debated for its variants in structure with its powerful multi-head self-attention mechanism on ECG signal for capturing important features for ECG analysis.

To explore the transformer networks for ECG analysis thorough research is required where the focus should be done on the variants of the structure of the transformer networks with its multi-head attention mechanism. Detailed work is required that can benefit the research community by filling the gap of research questions in the transformer networks for the task of ECG analysis and providing the list of potential future objectives. Therefore, this work aims to investigate about the following research questions:

- Which variant of the structure of transformer networks either encoder-only structure or encoder-decoder structure, are more effective and efficient in performing ECG analysis?

- Which factors are important for considering while designing the algorithm for ECG analysis using different variants of structure the transformer network?

## 1.3 Scope

The scope of this research study about the evaluation of using the transformer networks for ECG analysis is limited to the duration of four months and during this time, two variants of the structure of transformer networks that is encoder-only and encoder-decoder-together will be researched for ECG analysis and a maximum of thirteen thousand data samples from synthetic DeepFake ECG [40] will be used for evaluation of both variants models. These data samples are uncategorized i.e. dataset is not categorized into with or without diseases.

## 1.4 Research Methods

The research study done in this work will follow the paradigm of quantitative research methods where hypotheses about variants of the structure of the transformer networks will be formed, followed by the creation of models based on these hypotheses for ECG analysis. These models will then be examined for their statistical performance for ECG analysis based on their training on the DeepFake ECG dataset.

## 1.5 Ethical Considerations

AI-developments started right after the publication of Alan Turing's article "*Computing Machinery and Intelligence*" where he asked the question "*Can machine think?*" and proposed a method to find an answer which is called "*The Turing Test*". At the same time, started the development of controversies and ethical consideration complexities about AI's unpredictable results in almost all fields of science [22]

Ethical considerations for potential benefits from artificial intelligence consists of questions related to the safety of human being and the moral status of researcher and artificial intelligence (AI) tools. Medical ethics follow the fundamental ethical principles for the development and research work involving human subjects which comprises unharmful, beneficial, respect for autonomy, and justice [22].

Since this project work will come under the umbrella of medical science and computer science field, Therefore, it is important to consider both scientific research and medical ethics during the research of the transformer network for ECG analysis. Some of the ethical considerations for trustworthy research for medical tools that the participant of this work, and the team of research supervisors showed concern over and discussed during and even before starting the project are briefly discussed below:

- ***Privacy, protection and respect***

Information from the human participant should be confidential and protected from any security breach. Researchers and developers should consider themselves responsible for keeping the safety of such information about the research participant [22]. Ethical implications should be done to protect human dignity from any misuse of products such that they do not interfere with the fundamental rights and freedom of human beings. The Council of Europe has expressed the need for proper surveillance and governance over the developments within biology, medical science, that can protect the dignity, identity of all human beings and provide respect, freedom and integrity, and other human rights to everyone. In addition, this governance should also assess and identify the ethical questions that threaten human dignity and any potential misuse of the product [2].

- ***Reliability, fairness, trust, accountability and recoverability***

  An important consideration for researchers and designers is their collective responsibility for making the AI tool more user-centric which shall not pose any danger to its human user. If in case it consists of any possible or potential threat then the researcher is responsible for publishing the findings rather than hiding it and the amount of work should be done to rectify it [22]. If AI-tool or any medical research findings supersede the threat, then this should be the only way to continue with the product, but in any case sharing of information about findings is necessary that will generate a bond of trust between the developer and user. Another important aspect is the only positive intention for any research/development work and AI tools should not perform any un-intended jobs for example: storing patient information without their consent. Furthermore, the medical sector also knows and trusts the researcher for what the AI tool is intended for [22].

## 1.6 Main Contributions

The main contributions of this research work consist of following:

- Evaluation of the performance of two end-to-end algorithms for ECG analysis using the two variants of the structure of the transformer networks i.e. model-A with encoder-only of transformer network and model-B with encoder-decoder of the transformer networks. Both models take the single lead ECG data for analysis.

- Identification of reasons for the difference of components in end-to-end model for variants of the structure of the transformer network.

- Identification of gaps of research questions for ECG analysis using transformer networks.

## 1.7 Thesis Outline

The rest of this thesis report is organized as follows. In chapter 2, a detailed literature review is provided which will include the description of ECG analysis and transformer network architecture. In addition a summarized development of modifications in transformer network over ECG data is also provided in this chapter. In chapter 3, a formal description about the collection of information about ECG-dataset with its pre-processing and techniques used for the components of the implemented models architecture with its detailed implementation process will be provided. In chapter 4, results are presented which will be followed by the analytical discussion in chapter 5. Finally, a conclusive summary about the contributions made in this project will be provided in chapter 6.

# Chapter 2

# Background

The work in this project will start with the research of gathering knowledge about the topic of Electrocardiogram (ECG) and builds the foundation of it with the help of previous research work done in this field. A detailed discussion about the electrocardiograms will be provided, a detailed study of what it is composed of, and what results the health specialist can provide from it, will be discussed. Building of concept about the motivation and problem statement of this work where answers will find out about what automatic ECG analysis is and how is it helping health care clinicians and what type of relevant research work especially involving artificial intelligence has been done to improve the performance of automatic analysis of ECG. This chapter will mainly be divided into three sections. The first section will provide the background about the project where electrocardiograms, its analysis and available methods in the deep-learning field for ECG's automatic analysis will be discussed, then in second section, a brief introduction about the deep-learning framework "The Transformer Network" will be provided. The third section will mainly discuss the relevant research work done for ECG analysis and existing techniques involving the transformer networks.

## 2.1 Electrocardiograms (ECG)

The section of background is an important component of this research project and will provide the fundamental context of this work. It will deliver the overview of the topic of electrocardiograms and rationale research questions of electrocardiograms' automatic analysis using deep learning.

ECG is a medical test that records heart activity. It is the most common and cheapest medical test performed by the health professional to diagnose heart diseases and monitor the heart condition of the patient [23]. ECG test is done by the placement of up to 10 ECG electrodes on different positions in between the limbs and chest of

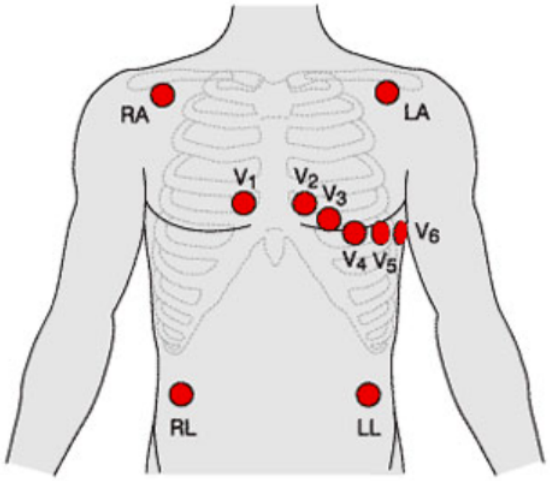


Figure 2.1: A picture of placement of ECG electrodes on human body (taken from [43])

the patient and generates 12 leads ECG graph. The process of capturing records are safe procedure and there is no risk of electric shock from the electrodes as it does not produce electricity rather it just records the electrical activity. Below a short description is given about the electrodes, leads of ECG, and what results are arrived after reading the ECG signals.

- **Electrodes of ECG**

  There is a difference between electrodes and leads. ECG electrode is defined as a conductor through which the current travels. These electrodes are stickers with wire connected to the machine which measures the heart activity in the form of an electrical signal and the system will record it and display it on the monitor or if nurses need can print it on paper too in the form of signals[10]. The electrodes only sense and pass the current through it to the machine while the leads are responsible for the representations on the machine in the form of a graph [23]. Following are the name of the chest electrodes [10]: $V_1$, $V_2$, $V_3$, $V_4$, $V_5$, and $V_6$ and the following are the limb electrodes: Red (RA), Yellow (LA), Green (LL) and Black (RL) as shown in figure 2.1 [10].

- **Leads of ECG**

  The 12-lead ECG is looking at the heart from 12 different locations and each one of them creates a slightly different graph. The first six leads of the ECG are called limb leads and are placed on the arm and legs and look at the heart from the right and left frontal planes from top and bottom [10] and are named: lead-I, lead-I, lead-III, lead-AVr, lead-AVI, and lead-AVf as shown in figure 2.2. The second six leads are called chest leads due to their placement on the chest of the

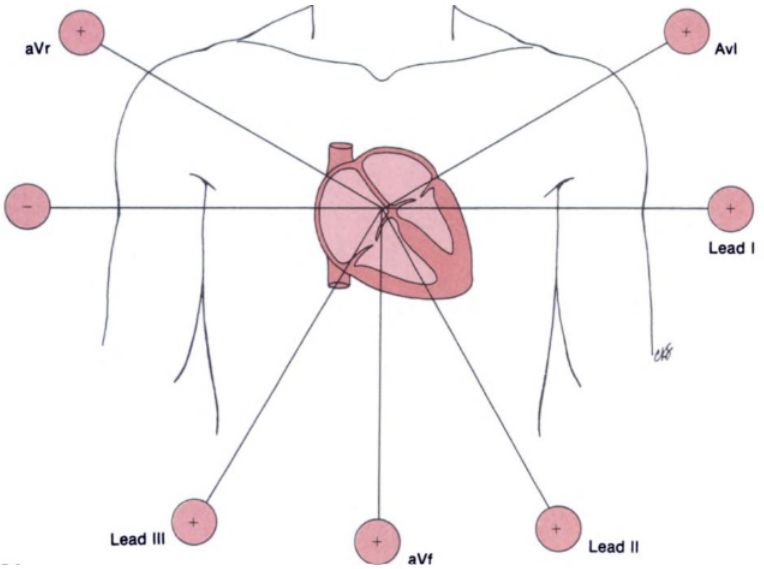


Figure 2.2: A picture of locations of ECG limb leads on human body (taken from [10])

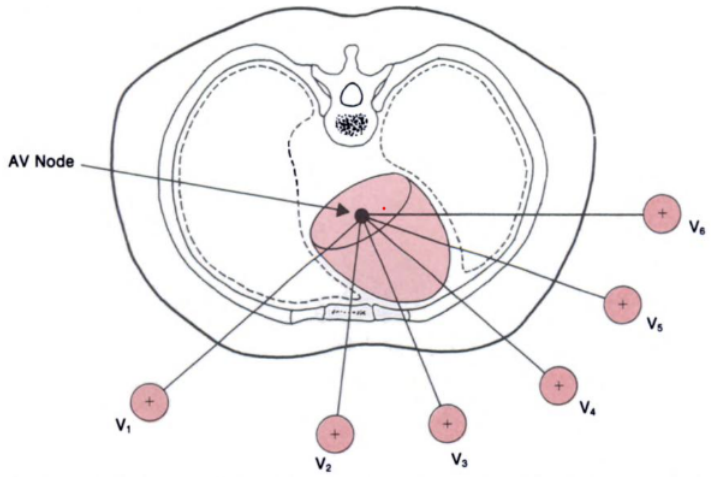


Figure 2.3: A picture of locations of ECG chest leads on human body (taken from [10])

patient and look heart from a horizontal plane [10] and are based on positive electrodes. They are named as V-leads: $V_1$, $V_2$, $V_3$, $V_4$, $V_5$ and $V_6$ and shown in figure 2.3.

- **Results from ECG**

  Health providers interpret the ECG wave signal and can provide for example following results or details about the heart.

  - **Heart rhythm** or **Heart rate** [35] is measured by measuring the pulse activity of the patient. Without an ECG device, it is difficult to calculate the irregular and too-fast pulse of the patient by the medical representative.

Therefore, ECG provides an overview of fast heart rate or slow heart rate. provides the information about the arrhythmia which is irregular electrical activities [16]. It can be read by the ECG by an irregular electrical signal.

- **Myocardial ischemia** [35] or in simple words **Heart attack** which happen previously or happening currently can also be determine by ECG by reading of the unusual previous or current signal waves of ECG.
- **Ventricular rate (Vent_rate)** is the rate of pumping blood by a ventricle of the heart which provides details about the Atrial Fibrillation [5] can easily be interpreted by reading the ECG signal.
- **Heart structure** which provides the heart position and its details about relative chamber size and any change to it in size will tell about the defect present in heart structure and can easily be read because of unusual electric activity by an ECG machine [35].
- **Electrolyte concentrations** [35]like blood and oxygen concentration plays an important role for any healthy heart and any changes to it can cause heart muscle failure and this also can be detected by ECG.
- **Drugs effects** [35] also changes the properties of a healthy heart and can be read through ECG signal.

To land on the above-mentioned results by the cardiologists from ECG, it is important to understand the ECG signal with its parts and its interpretations.

**Interval and Amplitudes in ECG signal**

A brief explanation of parts of ECG is provided below [3]:

- **P waves** interprets the atrial depolarization and in a healthy patient it should preceding to QRS complex.
- **PR intervals** is the duration of the start of the wave from P and ends at the beginning of the Q wave. This interpret as time taken between atria and ventricles.
- **QRS complex** represents depolarization of ventricles. It emerges from the three closely related waves on ECG.
- **ST segment** represents the time between depolarization and depolarization of ventricles. It appears at the end of the S wave and ends at beginning of T wave.
- **T wave** emerges as small wave after the QRS complex and represents ventricular re-polarization.

- **RR interval** is the duration of two peak of R waves.
- **QT interval** it appears from start of QRS and ends at T wave and represents time taken for ventricles to depolarize and then re-polarize.
- **R-peak Amplitude** refers as a the maximum amplitude of R point in QRS complex.

### 2.1.1 ECG Analysis

ECG analysis is the process of a detailed examination of the elements of the ECG signal. During the process the measurement of key elements that the clinician seems necessary for the diagnosis is done. These key elements from ECG are different amplitude at different points in the waves, different intervals between the particular points in ECGs, the shape of the waveform, duration of the different segments of the ECG wave, and the number of repetitions of the points in ECG.

ECG analysis requires a systematic approach, and for this one simply can make the procedure by following the approach of clinicians that how they identify abnormalities in ECG which are crucial for disease diagnosis and which part of the wave is important for mortality and morbidity of patient.

However, reading the long graphs of ECG signals for the above-mentioned relevant amplitudes and intervals is a hectic job for clinicians, and mostly ended up with an inaccurate analysis of ECG. Therefore the automatic analysis is done to remove the inaccuracy.

### 2.1.2 Deep-learning models for Electrocardiogram related tasks

Diagnosis of heart disease is a challenging task for doctors. It requires a tremendous amount of time to explore relative information from the electrocardiogram (ECG) records of the patients. Numerous cardiovascular diseases are diagnosed with the help of measuring key intervals and amplitudes which are in the ECG. These are hard to calculate with the naked eye and require time for that. Therefore, automatic analysis of ECG data is a hot topic of research in the previous couple of decades. The traditional and advanced methods are the two types of automatic analysis of ECG. In the traditional method, two steps are performed where first is to extract the extracted feature from raw ECG data with the help of the cardiologist, and then machine learning methods are applied to it to get the automatic results. Due to the human interaction in expert extraction, traditional methods of ECG analysis consumed a lot of human time and are still limited in terms of human expertise [17].

Recently, an advanced method of implementing deep learning over ECG data has given promising results because it does not require explicit extraction of features by an expert. This extraction is done by the deep learning algorithm automatically. Many research studies claim that deep learning models are powerful and flexible learnable tools with more informative extraction of features than expert extraction. The work presented by the team of [17] provides a list of multiple deep learning algorithms with or without desired modifications that are used for the classification, analysis, and denoising of ECG data or data imbalance challenges. Some of these deep learning algorithms are described below.

- Convolutional neural network is a widely applied deep learning algorithm in computer-vision, signal analysis and natural language processing field. CNN usually is the combinations of convolutions layer followed by batch normalization layer, nonlinear activation layer,dropout layer and pooling layer. It extracts the the pattern in data with the help of learn-able filter and kernel over it. CNNs have achieved good performance and faster computation due to the ability of parallization [17]. A little amount of pre-processing make it promising method against expert extraction in task of ECG analysis. Several algorithm have been proposed by the researcher for ECG analysis and diagnosis of heart diseases which solely utilizes CNN or have incorporated other deep-learning or machine learning method along with CNN.

- Recurrent neural network (RNN) is usually used for sequential data based task like time-series, natural language processing etc. Since ECG is also sequential and long length data therefore, RNN have been a preferred choice by researcher for capturing dependencies handling long varied data of ECG. Several RNN model and its advancement like Long-short term memory (LSTM), GRU have proposed by scientist where some uses attention module together with it in order to understand and visualize the attending location of model [17].

- Convolutional-Recurrent neural network (CRNN) is the combination of CNN and RNN and are used in ECG related tasks. 1-Dimensional CNN or 2-Dimensional CNN extracts the local feature from the segmented ECG signal. Then Bidirectional-LSTM (BiLSTM) picks up the global feature for better performance in classifications [17].

- Another type of deep neural network is auto-encoder (AE) where representation dimension is reduced by encoder and decoder tries to regenerate data from representations. Several variants of auto-encoder have been proposed like denoising auto-encoder(DAE), Sparse auto-encoder (SAE), and Contractive Auto-encoder (CAE) by researchers and are widely being utilized in denoising

the ECG signal. As introduced the earlier CNN, RNN, CRNN and BiLSTM are being used in either solely or in combination in encoder and decoder of these auto-encoder for better performance [17].

- Generative adversarial network (GAN) a famous framework of deep neural network where the first part is generative model *G* which generates the data similar to the original data distributions from a latent representation while the second part which is discriminative model *D* distinguishes between the real and generative data. IN ECG tasks, GAN are widely used to gear the imbalanced-data challenge. Data augmentation using GANs are extensively used in ECG data generation task [17].

Though [17] have listed most of the famous deep learning architectures with modifications still there were no information provided about the transformer network architecture with its modifications for ECG tasks. Similarly some of the other reviews about deep learning modules hardly mention Transformer network or its modifications used for ECG tasks.

## 2.2 Transformer Networks

This section of backgrounds will discuss the transformer network for developing the understanding of the techniques used inside in it.

In 2017, Vaswani et al. [44] presented the Transformer network architecture based on the self-attention mechanism which attend on dependencies to get the results. Transformer network takes the sequence as an input and performs parallel computing therefore it attracted the field of Natural language processing. This network outperformed sequence to sequence models in natural language processing like Recurrent Neural Networks (RNN) and Long-Short Term Memory (LSTM).

- **Input Embedding**

  Simply learned embeddings are done over input and output sequence in order to convert the token to vectors of dimension $d_{model}$. In [44] case, they put $d_{model}$ = 512. In layers weights are shared between them and in embedding layers those are multiplied by $\sqrt{d_{model}}$ .

- **Positional Encoding**

  To benefit from the order of sequence, some of the information about the position of token in the sequence must be injected to the transformer network. For this, positional encodings are done over input and output sequence at


Figure 2.4: Basic architecture of Transformer Network [44]

encoder and decoder stack respectively. [44] opted the learned positional encoding and using sine and cosine functions of different frequencies.

$$\begin{aligned} PE_{(\text{pos,2i})} &= sin(pos/10000^{2i/d_{\text{model}}}) \\ PE_{(\text{pos,2i+1})} &= cos(pos/10000^{2i/d_{\text{model}}}) \end{aligned} \tag{2.1}$$

- **Network Architecture**

  In natural language processing, transformer generates the output sequence when given input sequence $x$= $(x_1,\ldots.x_n)$. This input sequence is the sum of the embedding of each words in sentence and its positional encoding as shown in the figure 2.4.

  The transformer block is basically a encoder-decoder network where encoder maps the input sequence to the sequence of continuous representations $z$= $(z_1,.....,z_n)$. Given $z$, decoder then generate the output sequence $y$ =$(y_1,\ldots,yn)$ auto-regressively and one at a time as shown in figure 2.5.

  This encoder decoder is further divided into several identical number layers where the output of one layer is taken as input for the forthcoming layer. Figure 2.6 shows the layer architecture of the transformer model. For the ease of understanding, the further details of the architecture within the encoder and decoder will be discussed separately.

  - **Encoder:** The encoder is made up of several numbers of identical layers and each layer is further divided into two sub-layer which is multi-head self-attention system and fully connected feed-forward network. There is a residual connection around each sub layer which is added to the output of that sub-layer and then layer normalize before passing it to next step. Each sub-layer produces outputs into the same dimension (d_model) as of the

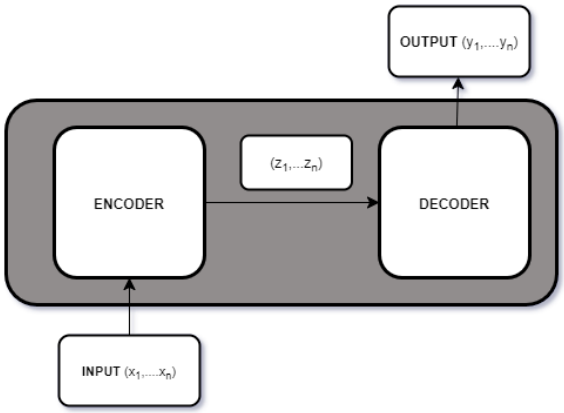


Figure 2.5: Transformer networks with Input, output and Attention sequence [44]

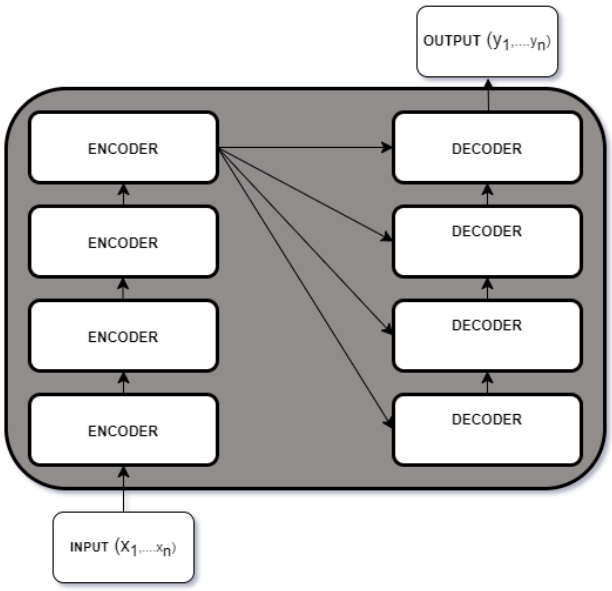


Figure 2.6: Transformer Network with Encoder and Decoder layers [44]

embedding and positional encoding. Dimension of the network, number of heads of the self-attention and the dimension of the feed forward network

is all tune-able and depends upon the requirements of the project.

- **Decoder:** Similar to the encoder, decoder also have stack of identical layers, with difference of three sub-layer in each layer. The additional third-sub-layer is placed between the multi-head self-attention sub-layer and feed forward sub-layer. This layer does the multi-head attention over the output of the encoder. Like encoder it also has residual connections to which the layer normalization is applied. A slight modification is applied to the self-attention sub-layer in decoder where masks are applied on position and subsequent position so that the system prevents the position for attending and made the prediction system depends only on the known output before it.

- **Attention:**

  Attention mechanism is most valued part of the transformer architecture which draws the global dependencies between input and output.

  * **Scaled Dot-Product Attention:**
    There are two most common attention functions in attention family. First is called additive function and second is called dot-product (dot mean multiplicative) attention. Dot-product is faster and more efficient than additive because it is using highly optimized matrix multiplication code. Vaswani et al. [44] suspect that if the large values of $d_k$, dot-product grows largely in magnitude and pushing softmax function to extremely small gradients. Therefore,the work in [44] include the dot-product with modification of adding scaling factor to it by multiplying it with $(1/\sqrt{d_k})$ and called "Scaled Dot-Product Attention" as shown in figure 2.7.
    Dot-product computation done on query with all keys with dimension of $d_k$ and $d_v$. Here $k$, $v$ are values and keys. Then divide each by $\sqrt{d_k}$, and apply softmax function over it to obtain weight on values [44]. The output can be computed as shown in 2.2.

$$Attention(Q,K,V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2.2}$$

  * **Multi-head Attention:** The number of head time application of attention function is beneficial than performing only single attention function to keys, values and queries. Therefore a multi-head attention is designed to allow model jointly attend on information from different representation at different positions. These are concatenated and projected which resulting in final values. This can be calculated as

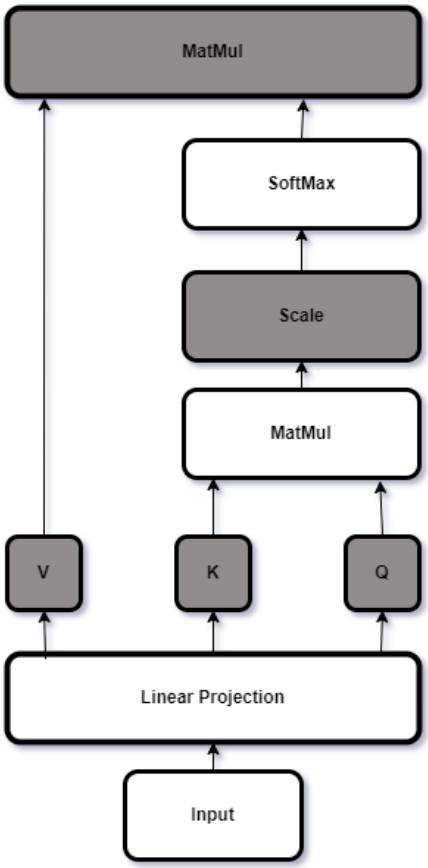


Figure 2.7: Structure of Scaled dot-Product Attention [44]

shown in 2.3 and illustrated in figure 2.8.

$$\begin{aligned} MultiHead(Q,K,V) &= Concat(head_1,..,head_{\mathrm{h}})W^{o} \\ where\ \ head_1 &= Attention(QW_{\mathrm{i}}{}^{Q}, KW_{\mathrm{i}}{}^{K}, VW_{\mathrm{i}}{}^{V}) \end{aligned} \tag{2.3}$$

## 2.3 Related work

This section will provide the information of existing methods for ECG analysis which develop the understanding of the research questions.

### 2.3.1 ECG analysis task using CNN

In this section we will understand the existing techniques of ECG analysis using CNN. the rest of the paper using transformer networks discuss in this work are used

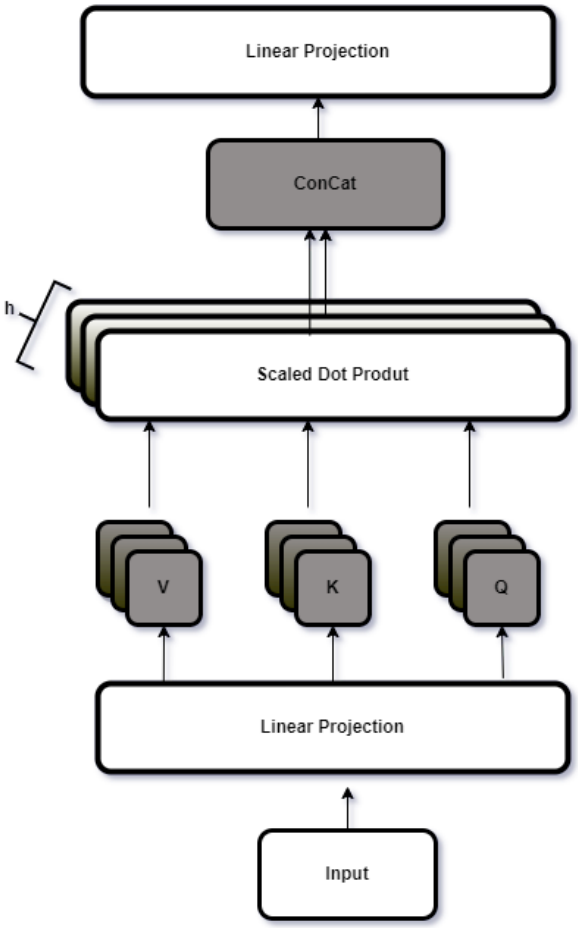


Figure 2.8: Structure of Multi-head Attention [44]

for classification techniques, therefore, this work is included in order to understand the task of ECG analysis.

**Explaining deep neural networks for knowledge discovery in electrocardiogram analysis [15]**

This work presented a residual convolution neural network-based regression study, which can quantify relevant interval (time dimension) and amplitudes (voltage dimension) from the ECG more accurately than doctors. The CNN predictions outperformed the cardiologist by the huge margin.

The model architecture consists of the blocks of convolution layer, batch normalization, ReLU activation. The ECGs are passed through these blocks to get the features and then passed through the average pooling layer. The obtained pooled features are then passed through the eight residual blocks which are composed of

two sequential blocks of convolution, batch normalization and ReLU activation. The features set from the last residual block are then average pooled to get the predictions of intervals and amplitudes. The prediction from the model includes PR intervals, QRS duration, heart rate, QT interval etc.

The model is trained and evaluated on either 10s raw 12-lead ECGs or 12SL-generated median beat from GESUS dataset [21] . The used the uncategorized dataset i.e. not divided into normal and abnormal groups. The metric used for evaluation are mean absolute error (MAE) which is easily interpretable, and root-mean-squared error (RMSE) as it is more sensitive to outliers. Along with the residual CNN model their novel finding includes visualization of the attention map of the model and classification of sex of the patient. They incorporate GradCAM approach [38] to model to visualize the features of ECG that are responsible for that prediction. This will make the model not only resulting reliable prediction but also explainable. For classification of sex task, their model visualized the QRS complex as a responsible feature.

The learning point from their study is that they used the un-categorized data-set. This is because since the model is predicting intervals and amplitudes present in ECGs so there is no need of data-set which are categorized into normal and abnormal groups. This also results in better interpretations of the result rather than limiting it to a set of categorized. Another knowledgeable fact from their study is that they also tested the model with an abnormal ECGs consist of the abnormal values of amplitudes and interval. This will make sure that the model is not only predicting average but calculating the relevant interval and amplitude either normal or abnormal present in the ECGs.

### 2.3.2 ECG tasks using Transformer Network

Recently, researchers of other fields are also attracted to use the transformer network due to its superior performance, in their respective field of research. Among the advantages of transformer, its ability to attend on long range sequence is especially attractive to time-series modeling tasks [46]. Several modifications have been made in the transformer network by the researcher in order to answer the challenges of time-series analysis. Since ECG's task also comes in time-series modeling domain, several modified transformer network models have been proposed by the researcher for this task. Though majority of them are classification task for diagnosing diseases a very little amount of work has been done for predicting the relevant intervals and amplitudes from the ECG. This section of the chapter will review some of the relevant transformer-based models which are used for ECG modeling task.

**A WIDE and DEEP Transformer Neural Network for 12-lead ECG Classification [30]**

Their work includes handcrafted and automated feature extractions using ECG segmented data in order to perform classification of heart diseases . They named handcrafted features as WIDE and automated feature extraction as DEEP Transformer Neural Network. For DEEP extraction, firstly they created an embedding network that extracts ECG information from single lead ECG waveform segment. Then, they placed the encoder stack to which the embedded filter is feed as an input sequence. This encoder output the attention tensor which then feed to the multi-label classification head. Their embedding network is set of series of convolution layers applied to ECG waveform, in order to extract latent space representation from the signal. These representation are then summed up with the positional encodings. The summed result is then feed up to the encoder module of their model.

For the backbone of our models, a similar approach is acquired for extracting the embedded features from single lead, 10-second length ECG waveform. One of our models utilizes only the technique of using only encoder like them which output embedded attentions. As our objective is to perform the prediction of the relevant intervals and amplitude [VentRAte, QT-interval, QRS duration, R-peak amplitude]. It is important to work on the fact that is encoder output enough for predictions? and why a decoder is needed if we are getting the good results from encoder?

A tutorial blog [14] has also followed similar method [30] and implemented its work in PyTorch framework where deep method was utilized with only one feature of WIDE method . Their work was helpful in understanding the implementing techniques of PyTorch.

**Gated Transformer Networks for Multivariate Time Series Classification [28]**

Their work presented the gating techniques over the standard transformer network for multivariate time series classification problems including ECG classification. They named their model as Gated Transformer Networks. The gating technique merges the features extracted from the special designed two towers framework of the encoder of the transformer network. Their method is to capture the key information in the both step-wise (temporal) and channel-wise (spatial) co-relations in time-series research.

In embedding of time-series data they applied fully connected layer with non-linear activation *tanh* then positional encoding is added to it to encode the temporal information. Simple transpose the channel and time axis as time series is fed to each encoder.

In step-wise encoder, they encode temporal features using the self-attention

with mask to attend on each point in all the channel by calculating the pair-wise attention weights among the time steps. The attention matrix is formulating on all time steps by the multi-head self-attention layer module called scaled dot-product attention. The rest of the mechanism of feed-forward, residual connection and layer normalization are similar as of the standard transformer.

The channel-wise encoder computes the attention among the different channel across all the time steps. They only include the positional encoding in step-wise encoder as there is no relative co-relation between the order of channel and the time series. Attention layers on all channels, capture the correlation among the channels across all time steps.

A gating mechanism that learns the weight of each tower is applied to the output of the two-tower transformer encoder. This is the fully connected layer with non-linear activation then concatenated followed by the linear projection to get another layer which is then applied with softmax function to get the gated weights for encoders. Each gating weight is attending on the respective encoder's output and packed as the final vector.

**Constrained transformer network for ECG signal processing and arrhythmia classification [11]**

This work presented an end-to-end framework for ECG signal processing and arrhythmia classification, which is based on convolutional neural network (CNN) where a transformer network is embedded to CNN in order to capture the temporal information of ECG signals. A new link constraint is introduced by them to the loss function to enhance the classification ability of the embedding vector.

A record of 12-leads ECG is divided into equal-length segments of ECG signal according to the window function size and step size provided in pre-processing phase. This 12-lead data is then feed to CNN with seven layers of convolution, each with different kernel size, convolution filter, batch normalization layer, active layer, and pooling layer to capture features with temporal information. These features are then sent to the transformer layer where they have used only the standard encoder architecture of the transformer with embedded size of the encoder is limited 256 and feed forward dimension to 1024. The embedded output from the transformer layers are then feed to the classification layers for multi-classification. This layer consists of linear layers and activation layers, which output probability of heart disease in patient.

**A transformer based deep neural network for arrhythmia detection using continuous ECG signals [19]**

Their work present a model for arrhythmia detection in ECG signals. Their model incorporated not only the encoder of the transformer but also the decoder. They tried ECG data on a modified DETR framework which is an end-to-end object detection framework based on the transformer. DETR [9] is famous modification of transformer network in computer vision community.

They proposed the CNN backbone which extracted the temporal features of the ECG signal where they have utilized customized inverted residual block proposed in MobileNetV2 [36] and Squeeze-and-Excitation module [18] in their CNN based backbone. These features are feed to the slight modified transformer where modification is done on the input to the decoder. Since in ECG there is no target sequence known therefore they fed Object Queries to the decoder is the positional encoded embedding of number of queries. the output from decoder contains the the number of queries object embedding. These are then forward to predictions heads which are the feed forward networks (FFNs) which results in classifying the heart beats in the ECG.

## 2.4 Summary

To summarize this chapter, a fundamental understanding of ECG is provided. A brief discussion about ECG analysis is delivered where the importance of accuracy in the analysis is mentioned and a discussion was done about how clinicians ended up with an error reading the ECG signaland talk about the significance of AI-aided ECG analysis was made. Furthermore, a detailed discussion about the superior results of the transformer networks is provided with information on its components. Lately, this chapter mentioned the related existing techniques which help in developing the models for analysis.

# Chapter 3

# Methodology

This chapter will describe the developments made in the project systematically to evaluate the usage of the Transformer network over the ECG dataset. First of all, a detailed description of the dataset used for evaluating the proposed methods of the project is provided and it is then followed by a detailed analysis of selected methods that are used in the creation of the architecture of the proposed models to extract information from the ECG dataset. Later, the documented development of the experimental setup and tools used for setting it up will describe which will help other researchers to reproduce and double-check the results. The code for reproducing the results are available at GitHub repository [13].

## 3.1 Dataset

In medical sciences, the production of good quality ECG data-set has grown increasingly and will continue to grow in the future. Since a couple of variables in the ECG data set stored the individual information of the patient. Even after anonymizing and de-sensitized it, a couple of private information can be taken out from the data set with the combination of multiple variables [40]. Therefore, due to privacy issues like General Data Protection Regulation (GDPR) in European Union (EU), the open-access availability and sharing of this good quality data set among the researchers is still a considerable problem. Synthetic data which carries similar information to real data, is the answer to the problem of privacy issues specially in the medical world where consent from the patients limited the availability of information. In this work, the ECG signals used for training and evaluating the proposed models are acquired from DeepFake electrocardiograms [40]. DeepFake ECGs are 10-second long, 12-lead ECGs that are created using generative adversarial networks (GANs). The developers trained the GANs with 7,233 real normal ECGs and produced 150,000 synthetic ECGs, out of which 121977 (81.3%) are classified

| Variables | Unit | Real-normal | DeepFake-normal | DeepFake-all |
|---|---|---|---|---|
| Heart Rate | BPM | 70 | 70 | 70 |
| QT Interval | ms | 105 | 117 | 118 |
| QRS Duration | ms | 90 | 92 | 93 |
| R Peak amplitude | µV | 1287 | 1275 | 1273 |

Table 3.1: The standard ECG parameters (mean) in real and fake ECGs. \textit{BPM} beats per minute.∼\cite{ecg-pulse2pulse}

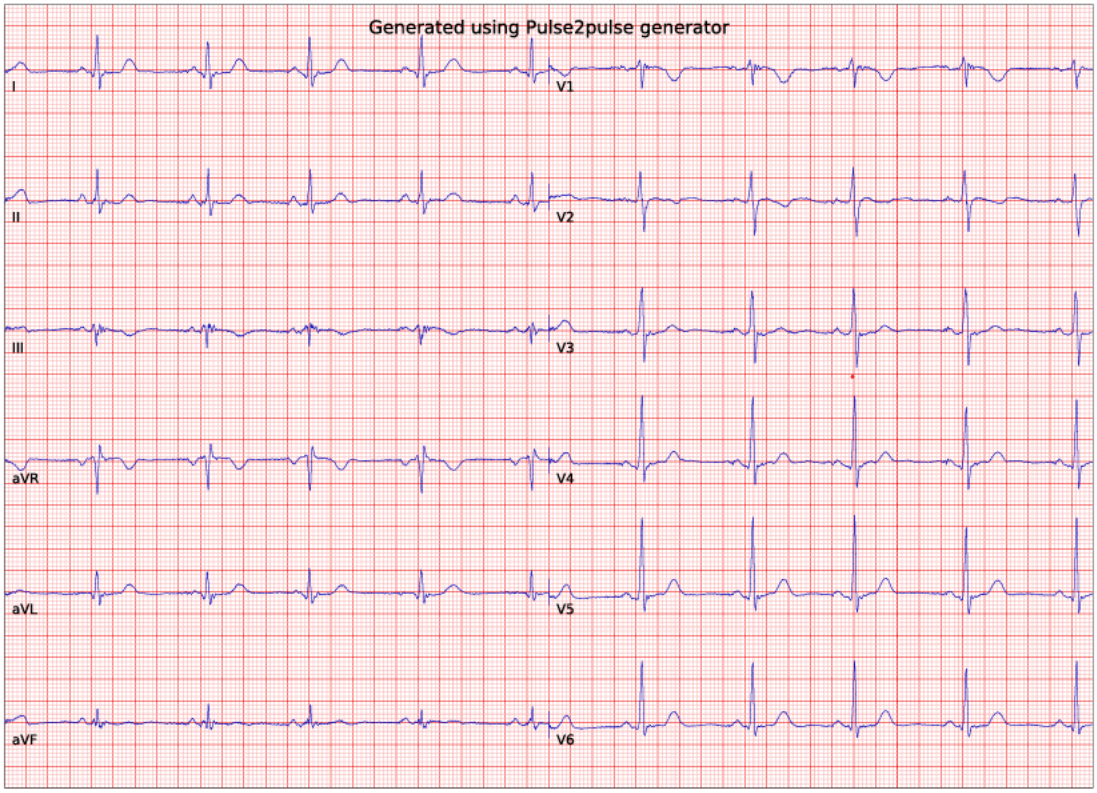


Figure 3.1: A sample from DeepFake ECG dataset (taken from [40])

as normal ECGs by commercial ECG interpretation program (MUSE 12SL, GE Healthcare). Amplitudes and intervals in synthetic ECGs were similar to the real ECGs. Despite the fact of being trained on real ECGs, DeepFake ECGs are not containing any data which are linked to the individuals and are therefore available for open access with its ground truth [12]. Table 3.1 shows the properties of DeepFake ECGs in comparison with the real ECGs which are used for its training. A sample of 12-lead ECG signal from the DeepFake dataset is shown in figure 3.1.

- **Data pre-processing**

  The proposed models of this work take the raw single lead ECG signal as an input to the model. For this purpose extraction of a single lead ECG signal

and ground truth for ventricular rate, QRS duration, QT interval, and R peak amplitude from the dataset is done in the data pre-processing step. In this work for the sake of the evaluation of models, only Lead-I from DeepFake are used for training and validation purposes. In the Python program, a function is generated for this extraction of lead-I and ground truths and also its transformation to tensor done to data because the deep learning framework this work follows is PyTorch (discussed later in this chapter), then both extracted lead data and respective ground truth are send to the data-loader of the training program where it splits into training and validation batches and load for training and validation program.

## 3.2 Proposed Model

Before presenting the model or discussing the methodology used for creating it, it is important to look again at the motivation and problem statement which will make it easy to understand the reasons to use the presented methods in the project. The main problem to be solved in this work can be mapped out as follows: Given 12-lead synthetic all normal ECG signal data, a single lead is extracted and passed through the model which incorporates the transformer network for learning the prediction of relevant intervals and amplitudes in ECG signal called ECG analysis.

This work proposed two end-to-end ECG analysis models and named them as Model-A and Model-B. Both models have three components in their end-to-end structure. First component is common in both models and called backbone which is processing the raw one lead ECG signal and output the embedded positional encoded sequence of ECG. Second component is slight different in both model, Model-A is using transformer encoder and self attention pooling while Model-B is using full transformer which means both encoder and decoder is used in it. Third component is also shared by the both models and called feed-forward predictor head. The simple overview of end-to-end structure is shown in figure 3.2. This section now first provide the detailed explanation of the shared components with their mechanism which is backbone and feed-forward predictor head. Later in this section Model-A's and Model-B's second component which is transformer networks part will be elaborate in detail with structure figure.

### 3.2.1 Backbone

After developing the concepts of transformer network in chapter 2, it is now known that input to the encoder of the transformer should be an embedded sequence with its positional encoding to obtain the full temporal features of self-attention from

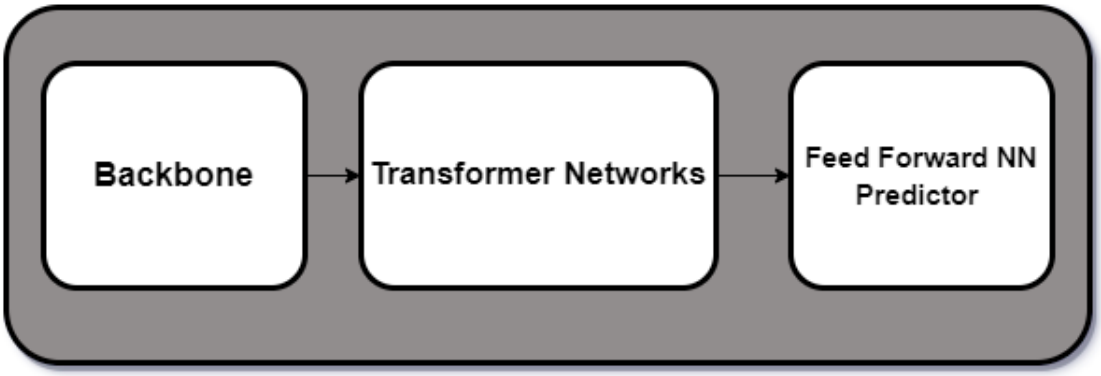


Figure 3.2: Overall structure of proposed end-to-end models for ECG analysis

the given sequence. In this work, the process of performing embedding of the ECG sequence and positional encoding of it will be known as a backbone to the model, which will provide the embedded- positional-encoded sequence to the transformer part of the model. Below is a detailed explanation of the methods which are used in the backbone:

- **Convolutional Neural Network (CNN) Embedding**

  In standard Transformer paper [44], input to the encoder is a word sequence where word embedding is done to it to get each word representation in the number form, so the further process of the transformer networks can be done. Since this work is applying transformer network on a synthetic DeepFake ECG sequence which is the digital representation of the electric activity of 10-seconds and is in the form of the number, i.e, it is a sequence of 5000 points represented in numbers, this work will apply the series of one-dimensional convolutions (1D-Convolutions) over ECG sequence to capture the latent space representation of the sequence. Results of 1D-convolutions methods are similar to word embedding where a scalar representation of a word about the other words in the form of sequence is created. The same goes here in 1D convolutions where different filters with kernel sizes are applied to the ECG sequence which does the element-wise multiplication then sums the results and gives the scalar results as the embedding sequence. It is also worth noting that the required output should be in the same dimension as the embedding dimension of the transformer networks, therefore, the last of the convolution layers the out-put channels are set to be similar to the embedding dimensions of the transformer (d_model in this work). Hence, performing the

| Conv.Layer | Input size | Output size | Kernel size | Stride | Padding |
|---|---|---|---|---|---|
| 1 | 1 | 128 | 3 | 1 | 0 |
| 2 | 128 | d_model | 3 | 1 | 1 |
| 3 | d_model | d_model | 3 | 1 | 0 |
| 4 | d_model | d_model | 3 | 1 | 0 |

Table 3.2: Convolutional layer configurations. d_model is the embedding dimension of the transformer network. In this work, it is set to *64*

one-dimensional convolution which element-wise multiplication and summing of result is a possible way of doing the embedding of ECG sequence.

In this work, the embedding of sequence in the backbone is inspired from [14] and it consists of a stack of four one-dimensional convolution layers (1D-CNN). The configurations for the convolutional layers used in this work are listed in table 3.2 where the number of kernel size, strides, and padding for each layer is defined. Each convolutional layer is followed by the rectified linear unit (ReLU) activation function which will output the input if it is positive. The last two layers of the convolution neural network contain a max-pooling layer after ReLU which does the pooling operation where maximum values are calculated from the feature matrix. The stacked layers of the convolution neural network for the embedding of the ECG sequence are shown in figure 3.3.

- **Positional Encoding:**

  According to [44], to make the use of order of sequence, it is required in the transformer network to introduce the information of the relative and absolute position of the embedding of the ECG sequence. Without the positional information, it will be difficult for the attention of the model to understand the order and might mix the semantics of the sequence which will result in bad performance of the model [24]. There are two places where positional encoding is required in Model-B where a full transformer with encoder and decoder is used. Since the procedure is similar to decoder input, therefore, this backbone section will only discuss the positional encoding on the embedded sequence to the encoder of the transformer. The positional encoding mechanism will have a similar embedding dimension as of transformer (d_model) so element-wise addition will be done between embedded sequence positional encoding to get the final positional-encoded-embedded-sequence which will be fed to the transformer. Instead of element-wise addition, concatenation can also be done to this but according to [44], they opted element-wise addition method because it benefits the computation with less requirement of memory and training time

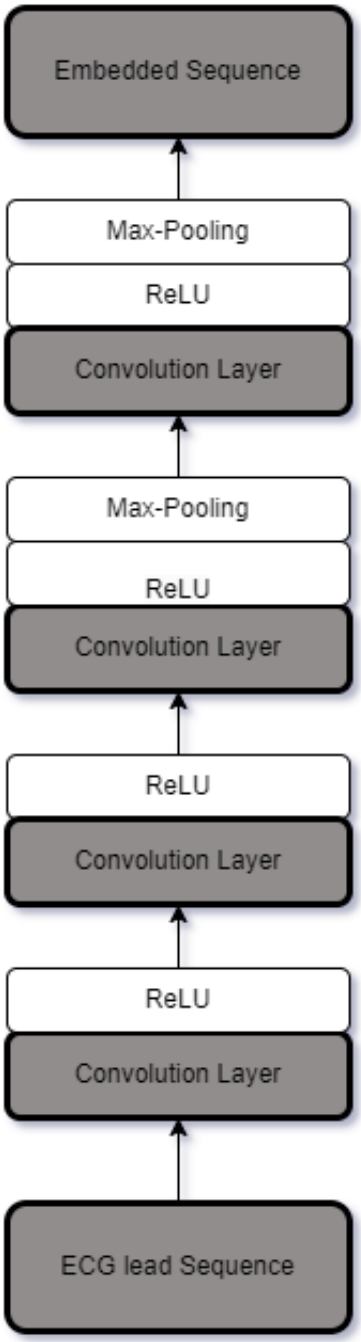


Figure 3.3: Convolutional layer Architecture

and does not require any further hyper-parameter. The element-wise addition of embedded ECG sequence and positional encoding is illustrated in figure 3.4.

Methods to encode the positions of sequence uses sine and cosine functions as

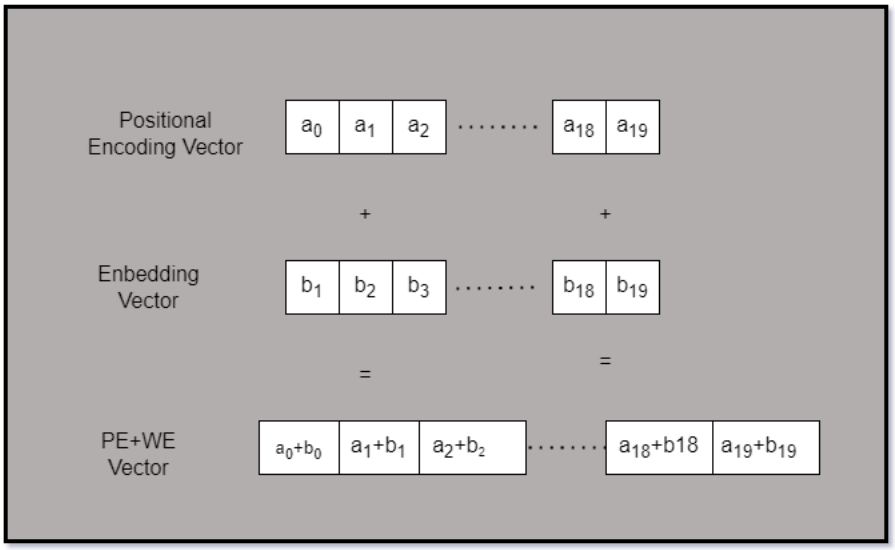


Figure 3.4: Positional encoding: element-wise addition illustration.

shown in equation 2.1 in chapter 2. For ease of understanding let's assume we have 20 points in the sequence instead of 5000 and the embedding dimension (d_model) is 64 then according to equation 2.1 each position is an integer from 0 to a max-sequence length minus 1 i.e., 0 to 19 with alternative sine and cosine values will end close to 0 and 1. The author of blog [1] has shown the visualization of positional encoding which make these functions more easily interpretable. For the example of 20 sequence length and 64 d_model, figure 3.5 visualizes the positional encoding of the example. Y-axis shows the sequence of each point whereas the X-axis shows the embedding and colors from dark to light (-1 to 1) are values of the positions of embedding. The positional encoder takes the input after a series of convolutions in the shape of [embedded-sequence, Batch_size, embedding-dimension] and outputs the sequence in the same shape of [embedded-sequence, Batch_size, embedding-dimension]. This output is now ready to feed into the second component of the end-to-end model which is the transformer networks but the detailed description of it will come with the description of the respective model later.

### 3.2.2 Feed-forward Prediction Head

The output from transformer networks either decoder or encoder is passed through the prediction head which consists of a feed-forward multi-layer perceptron (MLP). Multi-layer perceptron is a neural network consisting of a minimum of one hidden

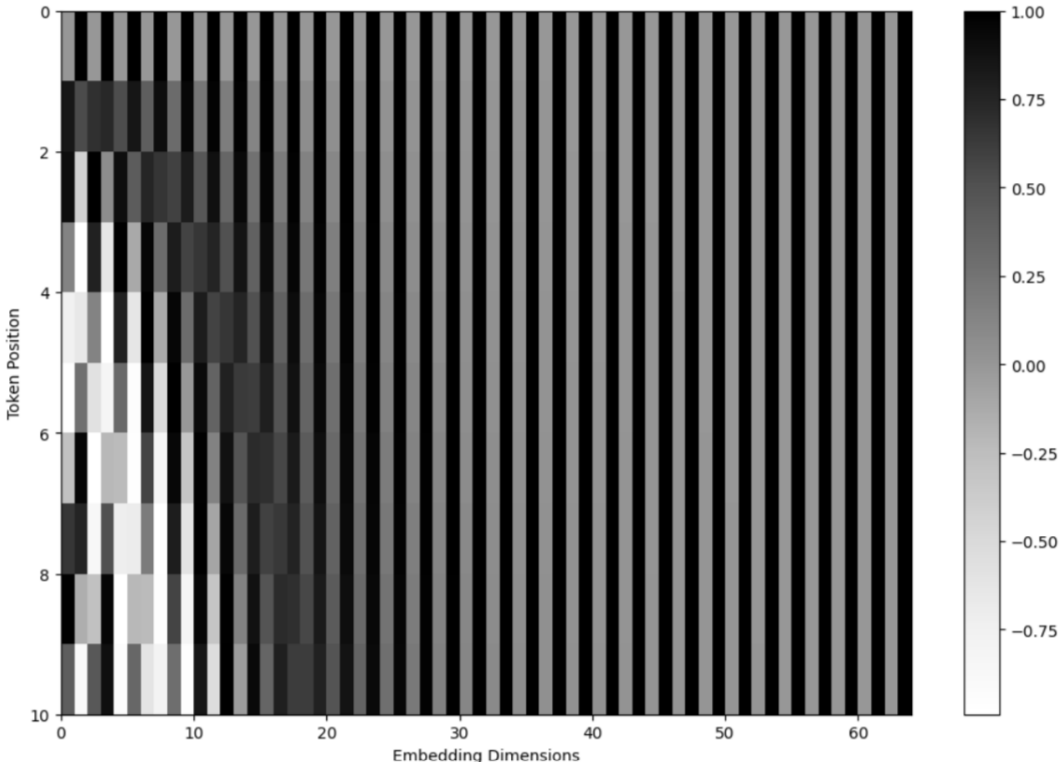


Figure 3.5: Positional encoding example illustration with sequence length *20* and embedding dimensions *64*

layer. MLP is considered flexible and is designed to learn representations from inputs. The neural network is often used for prediction tasks. In this work, it is also the requirement to predict the relevant properties of ECG from the attention sequence obtained from the transformer networks. Therefore A feed-forward MLP is used for this task. The prediction head MLP in this work, consists of eight linear layers each followed by the ReLU activation function, and the final layer is predicting a value for which the model is trained.

### 3.2.3 Model A-Transformer-Encoder-only

The proposed model-A is an end-to-end model which is using the first component as the backbone then the second component which uses a transformer network variant of encoder only and self-attention pooling mechanism, later it uses the third component where this model predicted the value for the relevant trained interval or amplitudes from ECG. The usage of only the encoder of the transformer is inspired from [14], [11], [30]. It is also important to find out whether the output of the encoder which is temporal attention of sequence is either good enough for prediction or not. The overall architecture of Model-A using only the encoder of the transformer in the second component is shown in figure 3.6. The detailed explanation of the mechanism

used in transformer encoder and self-pooling attention is described below:

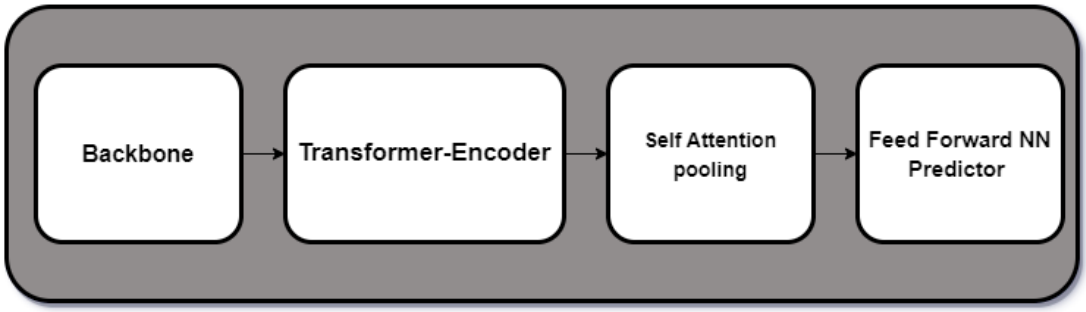


Figure 3.6: Overall Architecture of model-A with Transformer-Encoder-only.

- **Transformer Encoder**

  Embedded-encoded input with shape [embedded-sequence, batch_size, embedding-dimension] from the backbone enters the encoder of the transformer, where we have layers of the encoder, the concepts of layers within the encoder is already developed in chapter 2. Each layer of the encoder consists of two sub-layers of mechanism which are self-attention and feed-forward neural network. Self-attention looks into each point/token in sequence and allows it to relate with other points in the input sequence and create an understanding of other relevant points/tokens for the one to which it is currently under attention [1]. The first step of self-attention calculation will be the creation of three abstract vectors from each token in the embedded input which are called Query, Key, and Values with an embedding dimension of 64. These are created by multiplying the input token with initialized weight matrix which is later learned by the model during training. In the second step, the score is calculated for the considered token for itself and against all tokens in the sequence. This is done by the dot product of the query vector of this token with the key vector of itself and all the tokens in the sequence. By doing this it will end up creating the score for the token itself and relative scores for the other tokens in sequence. The third step will be dividing each score of the token with the square root of the embedding dimension of the key vector. Fourth step will be to apply the softmax function over it which will normalize it and explain how much each token expresses relation for that particular token in consideration. The fifth step will be to multiply the Value matrix of each token with each softmax score and then sum it up to get the attention matrix which has the little-bit relative information from all the tokens for the token in consideration [1]. The steps taken from first to

fifth are done to each token of the sequence provided to the encoder layer. Also, these steps are detailed explanations of equation 2.2 which is described in the scaled-dot-product section in chapter 2.

The author of [44] used the multiple self-attention layer by using multi-headed attention. The number of heads depends upon the requirement for the model and in this work, a total four number of heads are used after the inspiration of [14] work, to minimize the computational effort. This multi-head made refined the self-attention and give broad reference for every token relation and by this way we have not only one but four multiple randomly initialized weighted attentions and give different representation subspace i.e., four attention matrices [1]. Before sending these attention matrices to feed-forward neural networks, first, concatenate them together and multiply them with initialized weight which is then learned during training.

The second sub-layer of the encoder layer is position wise feed-forward Network where a fully connected feed-forward network consists of two linear transformations with different parameters between layers and ReLU activation in between is applied to each position of the token [44]. The dimension of the feed-forward layer is set to *512* for this project.

The residual i.e., skip connections is around each sub-layer of the encoder layer which is then added with respective sub-layer output matrices, and then layer-normalization is applied to it. These residual connections are the ones that make the positional encoding alive til the last of the mechanism in the transformer. Only the first layer of the encoder requires the positional encoded embedded sequence, the rest of the layers of the encoder (4 in this work, discussed later next chapter ) just take the output of the previous encoder layer as input. Finally, the output from the last layer of the encoder is the matrix of attention which contains temporal features of the ECG sequence and is ready to be entered into the next mechanism of the mode. The structure of the transformer encoder used in Model A is shown in figure 3.7.

- **Self attention Pooling**

  The output of the transformer encoder module is attention weights with shape [Sequence, Batch_size, Embedding]. A transformation is done to this shape before feeding it to the feed-forward prediction head. The transformation is a pooling mechanism that pooled out the attention from the tensor and makes it compatible with the prediction head. The author of [14] experimented with the two methods for this purpose, first is to take the average of the attention tensor and the second is to use the self-attention pooling layer over the attention tensor

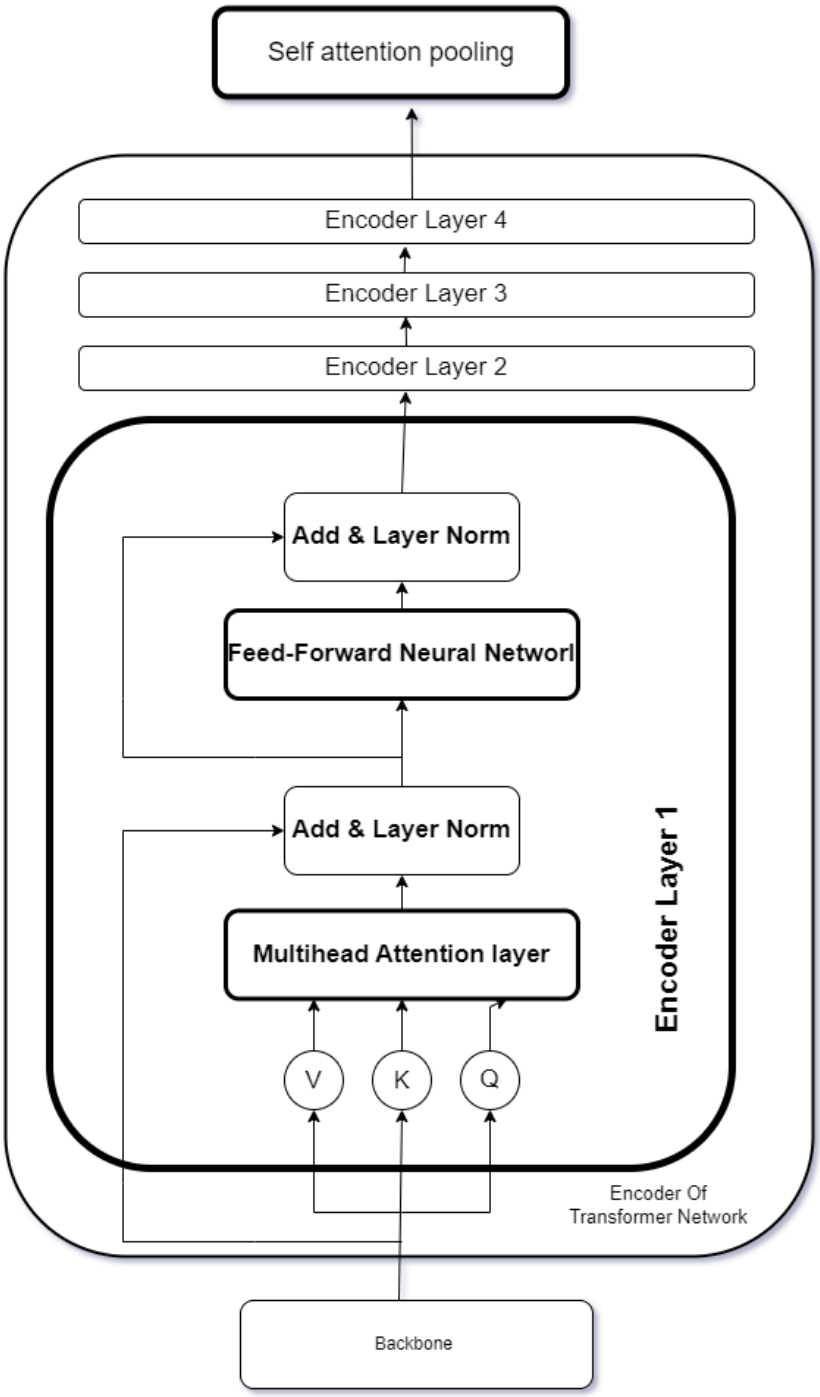


Figure 3.7: Architecture of encoder used for model-A.

and resulting in using the second method of self-attention pooling layer because it showed the better result than the average of tensor.

Self-attention pooling layer for encoder output was proposed for the research work of speaker recognition [34] but can be utilized in other cases. This layer

uses an additive attention mechanism that calculates compatibility using the feed-forward network with a single hidden layer. This self-attention is a dot-product where keys and values correspond same representation and the only query is need to be learned and trainable parameters. Therefore self-attention-pooling is a weighted average of the encoder sequence of features. This can be calculated by the equation shown in 3.1.

$$C = softmax(W_c H^T)H \tag{3.1}$$

Based on the research work of the team of [14], this work will also utilize the self-attention pooling layer to the output of the encoder of the transformer networks. The tensor of attention weighted sequence after self-attention-pooling is compatible with the next model of the end-to-end model which is the feed-forward predictor head and helps the model to learn better.

### 3.2.4 Model B-Transformer Encoder-Decoder

The standard transformer decoder [44] in the natural-language process is fed with the two inputs. The first input to the decoder is the target sequence with positional encoding to the first layer of the decoder. Since this project is dealing with ECG sequence and the goal is not to generate the target sequence nor target sequence is available in the ECG task. Therefore, an auto-regressive standard transformer decoder is not required for this project. Instead, a parallel computation of the target sequence should be done. Keep in the notice that no target sequence is either available therefore a technique is borrowed from detection transformer (DETR) [8] where the object-queries are created instead of the target sequence. It is the tensor of the object which is under consideration for prediction with a similar embedding dimension as of transformer. Embedding and positional encoding are done to it by passing it again to the backbone module of the proposed mode. So, in this model two times backbone is used, first for the ECG sequence's embedding and then for object queries' embedding. The second input is from the final output of the last layer of the encoder stack. The set of abstract Key and Values vectors from it. These abstract vectors are then fed as the first input to the decoder. The basic end-to-end structure of model-B is shown in figure 3.8.

- **Transformer Encoder-Decoder**

  The decoder is also the stack of several decoding layers as discussed in chapter 2.In this project, the number of layers is set to four. Each layer has three sub-layer where the first and third are the same as of the encoder i.e., the self-attention layer and feed-forward layer, while the second sub-layer is multi-head

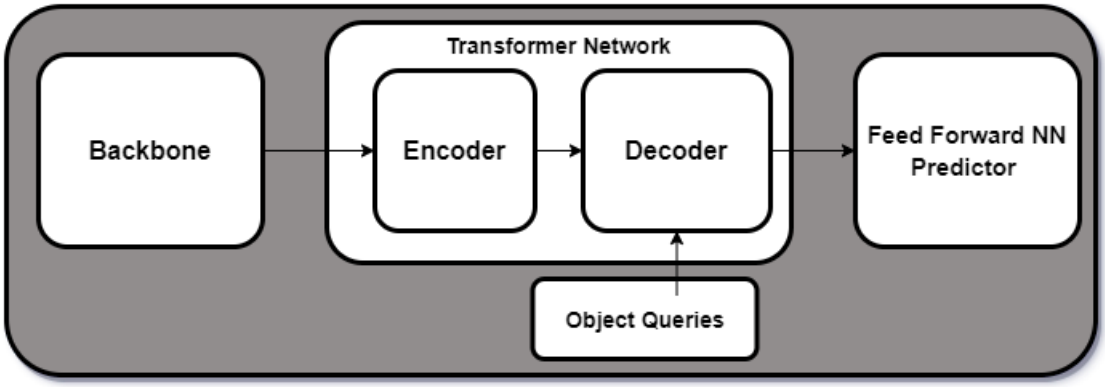


Figure 3.8: Overall Architecture of model-b with Transformer-Encoder-Decoder structure.

cross attention which takes the outputs of Key and values of the final encoder and from the previous decoding sub-layer of self-attention.

The first sub-layer of the first decoder layer performs the multi-headed scaled dot-product attention to embedded positional encoded object queries. The same five steps are done for the calculation of attention scores for each head and then concatenating each head score and multiplying with initialized weight to send it to the later sub-layer of the decoder layer.

The second sub-layer Multi-head cross attention is doing the same process that the multi-head attention layer is doing the only difference is that the inputs to it are from two different places. It takes the abstract Key and value matrix from the encoder output while the query matrix is from the previous attention sub-layer. Then it does the same procedure of calculating dot-product attention and sends further ahead for the feed-forward sub-layer.

The third sub-layer is the position-wise feed-forward layer which has the same mechanism and does the linear transformations to attention as described in the encoder of the model-A. Similarly, each sub-layer output is added with residual connection, and then applied layer normalization is over it to make the output ready for the next layer of the decoder. The rest of the three decoders do the same mechanism and finally output the attention weights of the ECG sequence. Another thing to notice is that the decoder always outputs the same shape as the object queries. Therefore self-attention pooling is not implemented this time and the output of the decoder is directly sent to the predictor module. The architecture of the decoder used in model-B is shown in figure 3.9.

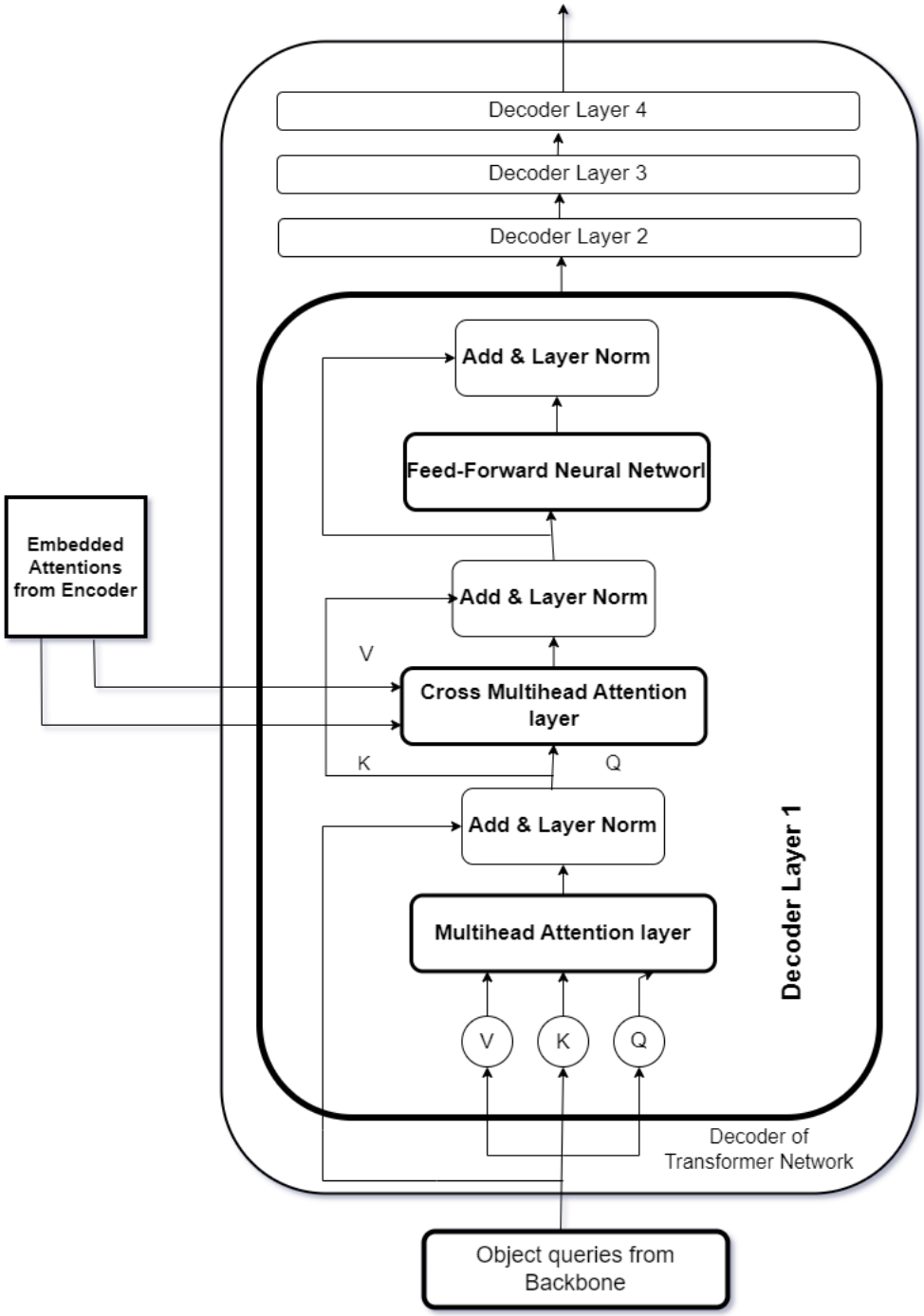


Figure 3.9: Architecture of decoder used in model-B.

## 3.3 Implementation details

This section of the chapter will describe the details of the implementation phase of the project which include the description of the tools used in the implementation of

methods and schemes for building, training, and validating the models.

### 3.3.1 Tools

This sub-section of the chapter will describe some of the tools used in the project during the implementation phase.

#### Deep-learning framework PyTorch

PyTorch is an open-source deep learning framework available with a Python interface and is known for its flexibility and simplicity. It provides excellent support for GPUs which makes it a popular choice for experimentation and building deep learning algorithms. This work used the PyTorch framework for building and experimenting with the models for ECG analysis. Some of the key benefits of using PyTorch are listed below [31]:

- Availability of large collection of built-in libraries.
- Excellent documentation of methods.
- Big, vibrant, and supportive community for discussion.

#### Colab

Colab is a cloud-hosted notebook from Google. It does not require installing or upgrade of your computer to meet the requirements of computational load for the training of AI algorithms [45]. In the early phase of training, only Colab was used for the training of models for different parameters of ECG, where a limited number of data units were available free of cost. Since this work requires an extensive computational load for multiple pieces of training, therefore, a monthly paid service of Colab-pro was used in the later phase of the project, where quite a few data units are available for monthly use.

#### GitHub

GitHub is a cloud-based service where developers can collaborate, show, store, and manage their codes for their projects in the form of repositories. This work also used GitHub as their collaborative platform for the development of the code for the project among the participant and supervisors.

### 3.3.2 Building, Training, and Validation Schemes for models

A Python code in the PyTorch framework is created where both training and validation loss is calculated in each epoch. First the parameters for training like the number of epochs, learning rate, batch size, and optimizer type for computing gradients of training were defined. Then the scheme for learning rate scheduler, early stopping, and saving the checkpoints were put in place. The data for training called for splitting from the pre-processing unit. Finally, iteration started for training and validation where the model is fed with the batches of data for ECG analysis and evaluation using loss-function. Following is a detailed description of the parts of the training and validation code with its components.

- **Loss function**

  The fundamental goal of every deep-learning algorithm is to improve the learning of the model. The learning process of the regression prediction task is inspected by using the loss function on the learning of the model. The goal is to lessen the loss between the target value and the predicted value. There are several metrics for loss function but in this work, the mean-squared error is used as the primary metric for the evaluation of the model training. During the phase of training, there was the need for better interpretation of model loss, therefore, mean absolute error was also calculated to easily understand the model training [20] but the loss function used for training was still mean squared error due to its maximum penalty for mistakes. In this work, the PyTorch library for measuring the mean square error and mean absolute error is used.

  - **Mean Squared Error (MSE)**

    L2 loss is known as squared error which is a difference between predictions and target values as shown in equation 3.2. While the cost function of squared error is called Mean square error as shown in equation 3.3. The majority of researcher prefers to mean squared error when it comes to the regression task, because of the squared penalty which is given more weight to the outliers and creates a smooth gradient for minor errors [20]. the MSE loss increases exponentially if models do mistakes in learning.

$$L2 = (y_{target} - y_{predicted})^2 \tag{3.2}$$

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y_i})^2 \tag{3.3}$$

– **Mean Absolute Error (MAE)**

L1 loss is known as absolute error which is the difference between predicted and targeted values as shown in equation 3.4. while the cost function of absolute error is called mean absolute error as shown in equation 3.5. MAE loss is simple to compute and robust and simple to interpret [20] because outliers may not strictly follow Gaussian [32].

$$L1 = \left| y_{target} - y_{prediction} \right| \tag{3.4}$$

$$MAE = \frac{1}{N} \sum_{i=1}^{N} |y_i - \hat{y_i}| \tag{3.5}$$

- **Learning rate scheduler**

  Learning rate is an optimization parameter and in this work, a scheme to tune it was used during the training of the model. If there is no improvement in loss for eight continuous epochs then a function for shrinking the learning rate to a specified factor is called during the training.

- **Optimizer** The job of the optimizer is to adjust and update the weight in the algorithm. it tries to adjust the parameters so that the model gets closer to minima. In other words, it is a process of finding the optimal parameters for minimum loss from the model. In this work, Adam [4] and AdanW [52] is experimented with as optimizer for model training. These are used from the PyTorch library in training code. There are several parameters like weights, learning rate, etc. that this PyTorch library of optimizer, tune automatically for the model training.

- **Early stopping**

  Similarly, When there are continuous twenty-five epochs where there is no improvement in loss then a scheme for early stopping triggers and stops the training program. Early stopping is a form of regularization, to save the model to move into the overfitting zone. A simple if-else condition of Python is used to implement the early stopping in the training program.

- **Saving checkpoint**

  During the training program if the epoch reaches the best loss it calls a function to save this checkpoint and names it Best-Checkpoint, where the optimizer setting is saved in a file so that it can be loaded again for the ECG analysis. If another best loss is obtained in later epochs then this file is changed with the new optimizer setting of the new epoch of best loss.

- **Model Implementation**

  For building the structure of models PyTorch framework was used. Where in the __init__() method, components were defined in the form of layers, whereas computations were done in the forward() methods. The implementations of components of the models are discussed below.

  - **Back bone**

    The backbone comprises CNN and positional encoding components. These were defined in the __init__() method of the model.

    * **Convolutional layers**
      1D convolution, ReLU activation, and 1D-max-pooling were done in the input sequence of ECG signal using the PyTorch library for 1D-convolution, activation functions, and 1D-max-pooling.
    * **Positional Encoding**
      For positional encoding of embedded sequence from CNN were done using a designated function that was borrowed from PyTorch Tutorials for the transformer network. [25]. After positional encoding, the embedded-encoded-ECG-sequence was ready to go into the transformer network.

  - **Transformer Networks**

    In the beginning phase of implementation, a code for the transformer network was developed to understand the requirements of every component of the transformer network. However, for the training and validation purpose, PyTorch built-in libraries were used and defined in the __init__() method of the model.

    * **Model-A (Encoder-only)**
      In model-A, PyTorch built-in library for encoder-layers and encoder-stack was used which output the embedded attention tensor which then sends to the self-attention pooling layer.
    * **Self-attention pooling layer**
      A function was created for self-attention pooling where implementation techniques were borrowed from [14]. This gives the output that is ready for and compatible with the predictor's head.
    * **Model-B (Encoder-Decoder)**
      Similarly in model-B, the built-in PyTorch library for full transformer is used and defined in the __init__() method of the model. However, a query sequence is generated of the same shape as the ground

truth. PyTorch's tensor generation method was used to create query sequences. The output of transformer networks directly sends to the predictor's head.

- **Feed-forward Prediction Head**

  A sequential container of PyTorch framework for the neural network was used for the prediction head and defined in the __init__() method. A total of eight linear layers of neural network with ReLU activation in between them were used for the prediction of value for analysis.

## 3.4 Summary

To summarize this chapter, two end-to-end models were designed for training on the synthetic DeepFake dataset. The model consists of three components, first is the backbone, which used the CNN and positional encoding techniques in it, and second component is the transformer which is different in both models. The first model is using encoder-only variant and self-attention pooling layer while the second network is using encoder-decoder variant and the target source is designed for the decoder of this model. The third component is the feed-forward neural network predictor head which predicts the parameter value from the given attention from the transformer networks. The whole project is implemented using the PyTorch framework and the code is available at [13].

# Chapter 4

# Experiments and Results

At this stage of the project when the implementation of codes for extraction of leads and ground truth, data loaders, model code, and training and validation program of both Model-A and Model-b is ready then it is time to verify the mechanism and solutions presented in 3. This chapter will provide the quantitative details of both models' training and as well as describe the journey of loss function from the point of view of learning of models. Below are the subsections that will provide the number of experiments taking place during the project duration for the evaluation of the transformer networks in terms of using them for ECG data.

## 4.1 Experiment 1 (Configurations of Transformer network parameters)

Firstly, an experimental setup is arranged for the configuration of the transformer networks' parameters where the number of layers of encoder and decoder, number of heads of self-attention sub-layer, embedding dimensions, dimensions of feed-forward sub-layer in transformer network is all tuned according to the resources available to run this project. In this project, a trial of standard configuration from the original paper [44] for the configuration of transformer network parameters was done first. Standard configurations are shown in the first row of the table shown 4.1.

Since the resources for the project are limited especially in the computational power aspect, due to this, the first trial with standard configuration gave the out-of-memory issue. Batch sizes for running the model are also playing an important in this configuration of transformer networks issue, so the continuous decrease in batch size was done but still, the result was the same i.e., out of memory. For troubleshooting the issue, continued reduction in above mentioned standard transformer parameters in combination with batch sizes was tried as shown in table 4.1, then the trial with the suggestion from the author of [14] for transformer configuration were done and

| Trials | nheads | layers | d_model | ffdm | batchsize | result |
|---|---|---|---|---|---|---|
| 1 | 8 | 6 | 512 | 2048 | 128,64,32,16 | limitation error |
| 2 | 6 | 6 | 512 | 2048 | 128,64,32,16 | limitation error |
| 3 | 4 | 6 | 512 | 2048 | 128,64,32,16 | limitation error |
| 4 | 6 | 4 | 64 | 2048 | 128,64 | limitation error |
| 5 | 4 | 2 | 64 | 512 | 32 | successful |

Table 4.1: Trials for configuration of transformer network and batch sizes

| Variables | **AdamW-optimizer** | **Adam-optimizer** |
|---|---|---|
| **Learning rate** | 1e-4 | 1e-4 |
| **Epochs** | 12 | 12 |
| **Learning rate decay (times)** | 1 | 0 |
| **Checkpoint epoch** | 6 | 12 |
| **MSE loss** | 55.12 | 46.2 |

Table 4.2: Configuration of optimizer for training.

the result was a successful run, which is shown in the last row of the table 4.1. These configurations are used in this project throughout the multiple experiments of Model-A and Model-B.

## 4.2 Experiment 2 (Configuration for training parameters)

The second setup was arranged to set the parameters of training, first, the setting of the learning optimizer is done. After the valuable information obtained from [14] about the configuration of transformer parameters, their utilization of the AdamW optimizer is experimented with a training session of 12 epochs, for the prediction of ventricular rate with 13000 training data samples and learning rate set to 1e-4. This experiment is then compared with another experiment where the Adam optimizer is used for similar training. The results are shown in table 4.2. The training session of AdamW was with several no improvement epochs that trigger the decay of learning rate, so there was 1 decay in learning rate in just 12 epochs training session, while the Adam training was quite stable and continuous decreases in error were noticed. Therefore, after this experiment, Adam is selected for the rest of the experiments in this work.

Secondly, for the setting of the rest of the parameters of training like loss function where mean-square-loss (MSE) loss is considered as a loss function for

| **Training Parameters** | |
|---|---|
| Training Epochs | 100 |
| Batch size | 32 |
| Optimizer | Adam |
| Learning rate | 1e-4 |
| Learning rate decay (no improvement in epochs) | 8 |
| Early stopping (no improvement in epochs) | 25 |
| Loss function | MSE |
| **Transformer Parameters** | |
| Number of heads (nhead) | 4 |
| Number of layers | 4 |
| Embedding dimension (d_model) | 64 |
| Fees-forward-dimension (ffdm) | 512 |

Table 4.3: Parameter for training for rest of the experiments

evaluating the learning of the model. MSE helps understand the model performance over the data samples and this project wants to focus on decreasing the larger error therefore MSE is best for training. Later, an inclusion of calculation mean-absolute-error (MAE) of the learning is done because it provides the interpretation of the loss journey during training. Similarly, learning rate decay and early stopping are set upon the number of no-improved epochs where there is no improvement in the loss. After a couple of tries, early stopping and learning rate decay are set for consecutive 25 and 8 no improved epochs respectively. The overall configured parameters for experiments are shown in table 4.3.

## 4.3 Experiment 3 (Self-attention Pooling layer)

The component of model-A is using transformer encoder and self-attention pooling layer to the output of the encoder before sending it to the predictor's head. While the transformer network of model-B is not using the self-attention pooling layer.

The reason behind using or skipping the self-attention pooling layer is the length of the sequence inputs to the transformer network. For model-A, the encoder

| # | Shape after encoder | Shape before Predictor | Shape after predictor | Ground truth shape |
|---|---|---|---|---|
| **With Self-attention pooling layer** | [8, 1248, 64] | [8, 64] | [8, 1] | [8, 1] |
| **Without Self-attention pooling layer** | [8, 1248, 64] | [8, 1248, 64] | [8, 1248, 1] | [8, 1] |

Table 4.4: Ablation experiment of self attention pooling layer for model-A

output of embedding attentions is used for prediction purposes and input to it are long embedded-positional-encoded sequences from the backbone, therefore, the output from the encoder will also contain the long sequence of embedded attentions. By passing this long-length sequence to the predictor's head, it will predict a single value for each sequence. So the shape of the final output from model-A will not be the same as ground truth which is a single value of the feature, due to which the performance of training will be doubtful. Therefore a method is required to extract useful attention and make it compatible with the prediction head. Therefore, the self-pooling layer is used to pool out embedded attention from the output of the encoder of model-A.

While the output from the decoder of model-B is the same as the target sequence the decoder carries no similarities with the shape of the encoder of the network. The target sequence provided to the decoder consists of the same shape that the predictor is expecting from the transformer network, therefore no self-attention is required to solve the compatibility issue here.

For verification above method of using self-attention, two experiments were carried out to check the output from the model-A network before starting the training. The first experiment is with the self-attention layer where the output shape after the prediction head is the same size as the ground truth which is shown in the last two columns of the second row of the table 4.4. The second experiment, where ablation of self-attention pooling layer is done to the model-A network resulted in differently shaped output from the predictor's head and is not compatible for evaluation with ground truth as shown in the last two columns of the last row in the table 4.4.

## 4.4 Experiment 4 (Training of Model-A Transformer-Encoder Only for Ventricular rate)

Following the configuration of parameters for training and experimental test about self-attention pooling layer usage in model-A, an experiment was done to evaluate the performance of model-A. In this experiment, model-A is trained with all normal DeepFake datasets to predict the ventricular rate (Vent_rate) of the ECG.

The hyper-parameter settings are done similarly to the parameters as shown in table 4.3. Total *13000* samples of data are used for this experiment from the DeepFake ECG dataset, out of which *90%* is used for training purpose of the model and the rest of the *10%* is for the evaluation purpose of the model-training.

Mean-squared error (MSE) is used as described above as the primary metric to evaluate the training of the model. The experiment of the training baseline model went well for *100* epochs and within the very first epoch training loss and validating loss went to *2344.25* and *55.479* respectively. Training and validating loss continues to decrease during the epochs. A couple of no improvements were also noticed during the training but it did not trigger the decay of the learning rate scheduler which was set during the implementation of the training program as discussed in 3. The training reached the best checkpoint at the epoch *87* where the training and validating loss were *0.32* and *0.63* respectively. Figure 4.1 illustrates the journey of training loss while figure 4.2 shows the downward ride of validation loss during the training. In the figures, the outliers are skipped to focus on lower values for a better understanding of how the training journey was for the model-A where it learns to predict the ventricular rate from provided ECG dataset.

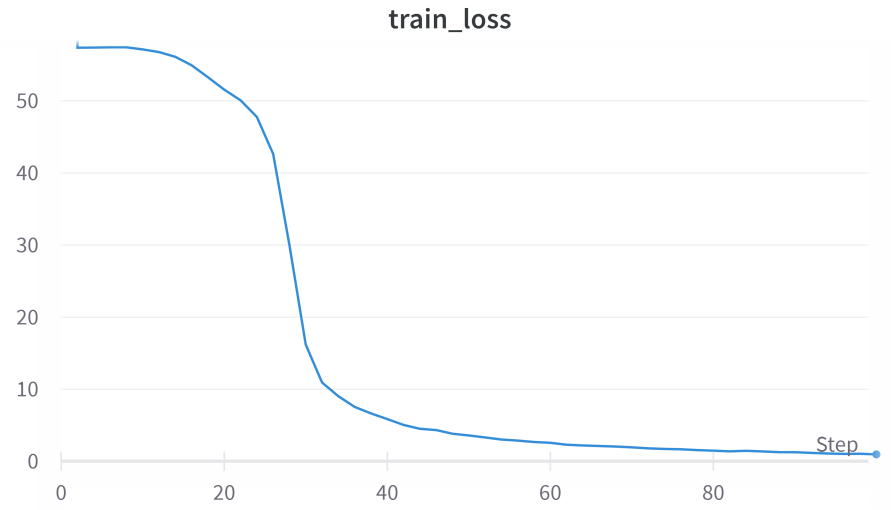


Figure 4.1: Training MSE Loss for model-A experiment

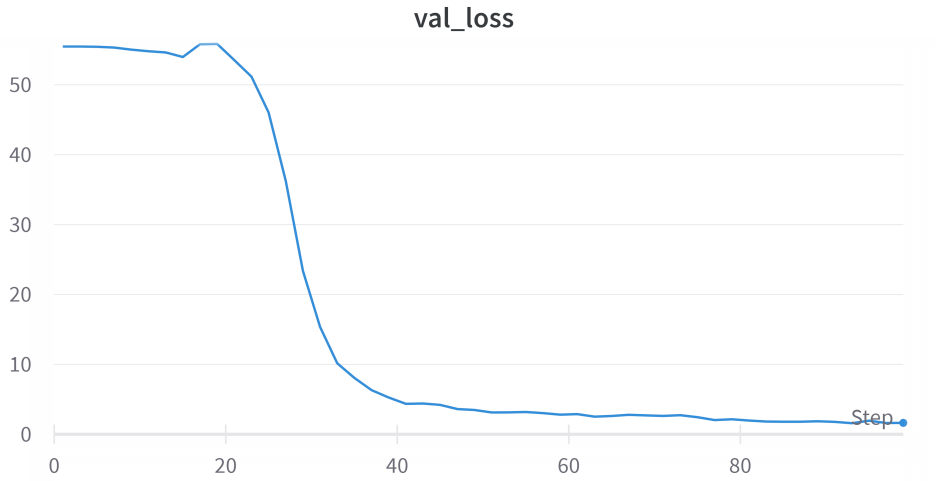


Figure 4.2: Validation MSE Loss for model-A experiment

| Variables | Training Samples | Validation error MSE | Zero-R error MSE |
|---|---|---|---|
| Vent_rate (BPM) | 13000 | 0.63 | 56.63 |

Table 4.5: Validation error and Zero-R error on DeepFake ECG (Encoder-only), training samples 13000.

This is the time when it is best to evaluate the performance of the model. Since there is no known research work available doing a similar task of ECG analysis where DeepFake ECG was used, a comparison could be done. Therefore, it is important to understand whether the results gets from model-A training is predicting actual values for Ventricular rate from signal or if it just has been learned to predict the average Ventricular rate (Vent_rate) from the provided training samples. Table 4.5 shows the evaluation of model-A of the Transformer Networks with Encoder only where it beats the ZeroR-estimate (predicting the mean).

## 4.5 Experiment 5 (Training of Model-A with limited data samples)

After the experiment for evaluation of the performance of model-A-encoder-only with 13000 data samples for predicting ventricular rate, another experiment was done to evaluate the performance of the model on the limited data samples due to limitations of resources like GPU and time consumption. In this experiment, the model is trained with the DeepFake dataset to predict the ventricular rate (Vent_rate), QRS duration (QRS_duration), QT interval (QT_interval) and R peak amplitude

(R_amplitude) features from the ECG signal. Four different training sessions were done with limited data samples as model-A is predicting one value at a time.

Configuration of the model for training is done similarly to the previous experiment as shown in table 4.3. Total 5000 samples of data are used for this experiment from all normal DeepFake ECG datasets, out of which 90% is used for training purposes of the model and the rest of the 10% is for the evaluation purpose of the model-training. Extraction of ground truth for all four predicting values is done similarly as described above in the implementation section of 3

For the training session of the second experiment, again mean-square-error (MSE) is used as the primary metric to evaluate the model with limited data but the calculation of mean-absolute-error (MAE) for batches and epoch is included in this experiment because MAE is easily interpret-able as discussed above. Details of training are briefly explained below:

- Training for ventricular rate with a limited dataset has a different journey as compared to experiment 1 where training samples were 13000. Within the few first epochs' training, MSE loss and validating MSE loss went to 56.27 and 51.85 respectively followed by a couple of no improvements which trigger the learning rate decay. Training and validating loss still did not start to decrease after decay and model showed the continuous no improvement in learning and this triggered the algorithm of early stopping which was set to 25 epochs of no improvement as discussed in chapter 3. the process of training went to just 36 epochs. The training reached the best checkpoint at epoch 11 where the training MSE-loss and MAE-loss were 56.27 and 51.85 respectively. Figure 4.3 and figure 4.4 illustrate the journey of MSE and MAE validating loss respectively.

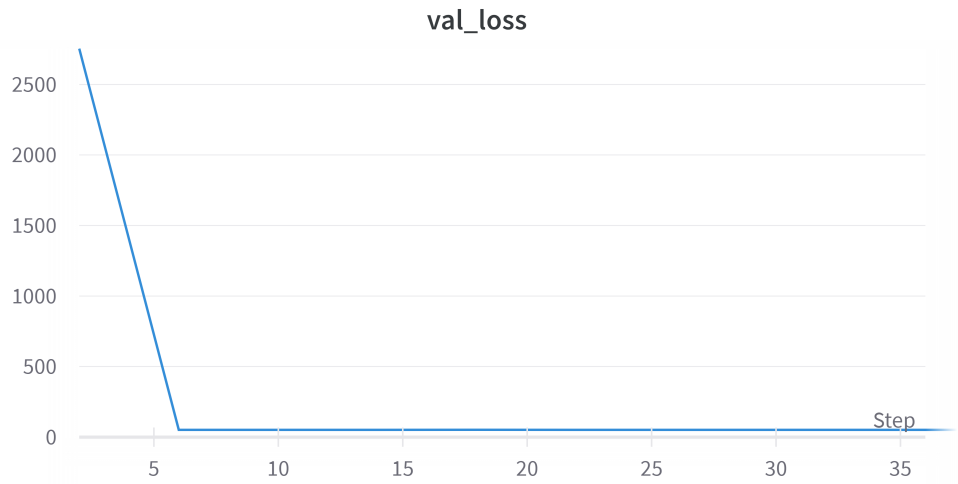


Figure 4.3: Validation MSE Loss (Vent_rate) for model-A training with 5000 data sample

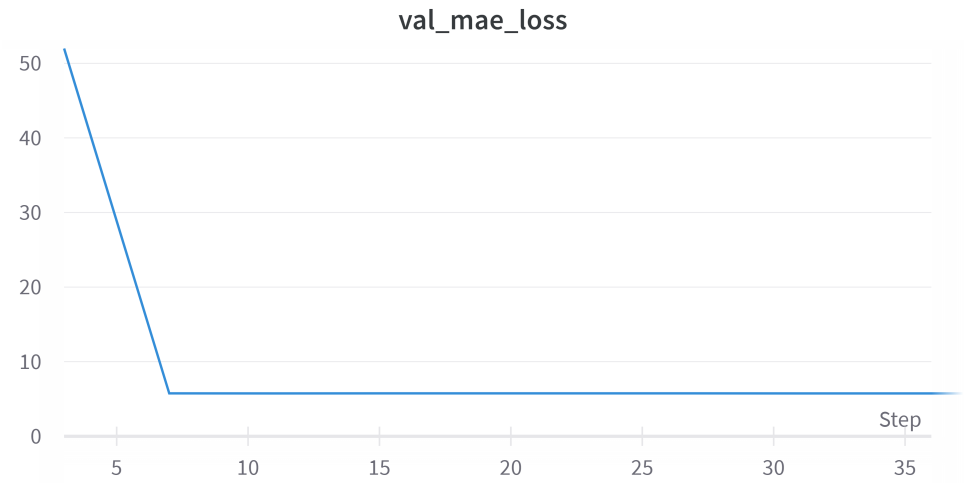


Figure 4.4: Validation MAE Loss (vent_rate) for model-A training with 5000 data sample

- For the prediction of QRS duration with 5000 training samples though training algorithm did not trigger the early stopping and went til 100 epochs but it did trigger the learning rate decays around seven times during the training process. Training function loss which is MSE loss dived from 8520.25 in the first epoch to 74.62 in the second epoch. It continued to decrease with a couple of learning rates decays as discussed before to a checkpoint point stage at epoch 97 where the best MSE loss was 69.15 and MAE loss was 6.89. Figure 4.5 illustrates the story of validation MSE loss while figure 4.6 shows the validation MAE loss of training for predicting QRS_duration from provided 5000 lead-I synthetic ECGs.

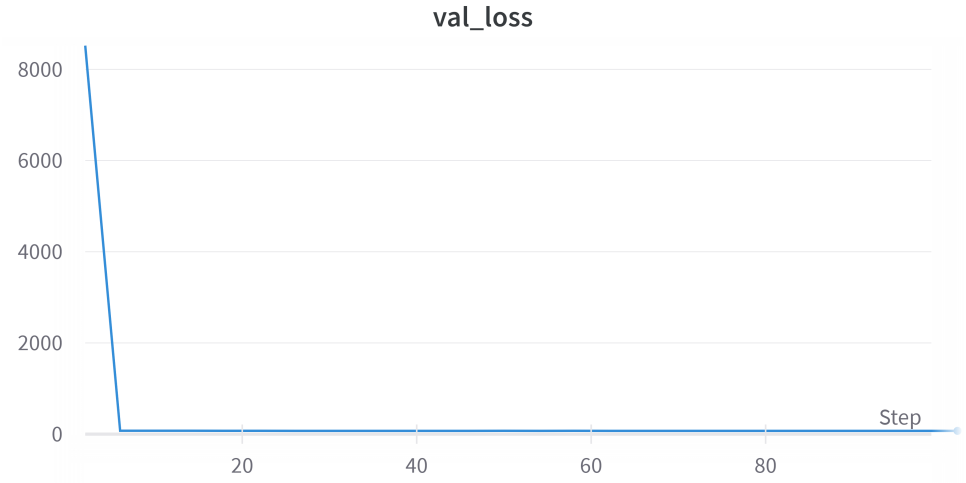


Figure 4.5: Validation MSE Loss (QRS_duration) for model-A training with 5000 data sample

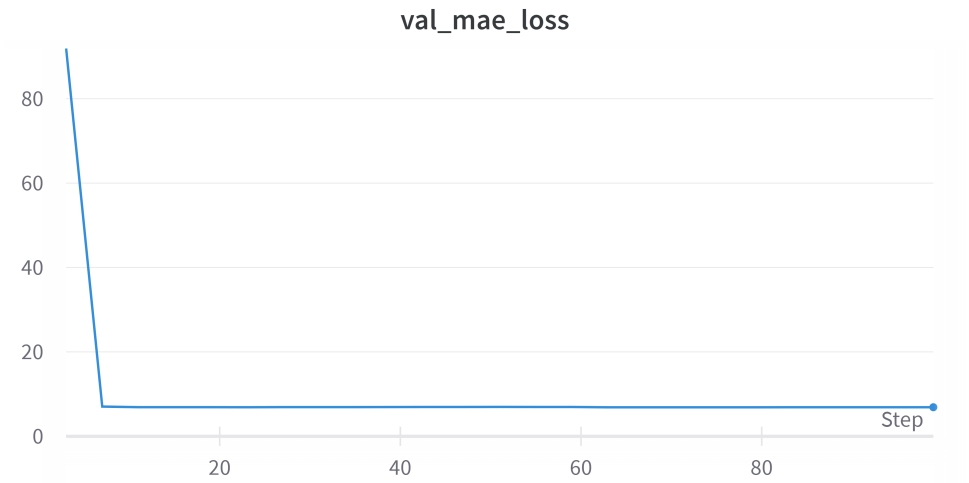


Figure 4.6: Validation MAE Loss (QRS_duration) for model-A training with 5000 data sample

- Training for QT_interval with 5000 training data samples had couples of no improvements but it only trigger once the learning rate decayed during the process of training with no early stopping to the 100 epochs. The best checkpoint reached at 94 epoch where training function MSE loss was 298.5 and MAE loss was 16.5. Figure 4.7 shows MSE loss and figure 4.8 shows the MAE loss graph for the training process.

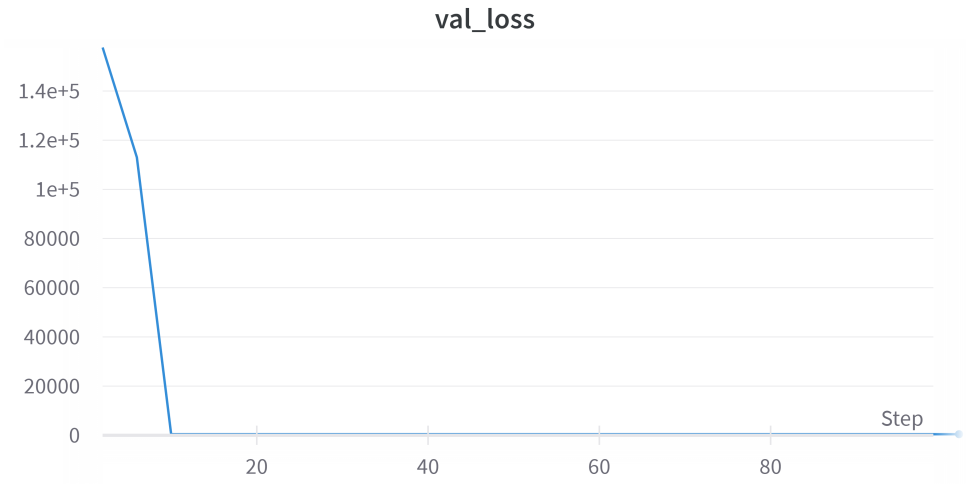


Figure 4.7: Validation MSE Loss (QT_interval) for model-A training with 5000 data sample

- The process of training for the prediction of R_amplitude with limited data started the decrease of function loss MSE with 521493 at the first epoch to 61083 at the second epoch. A couple of series of no improvement in function loss

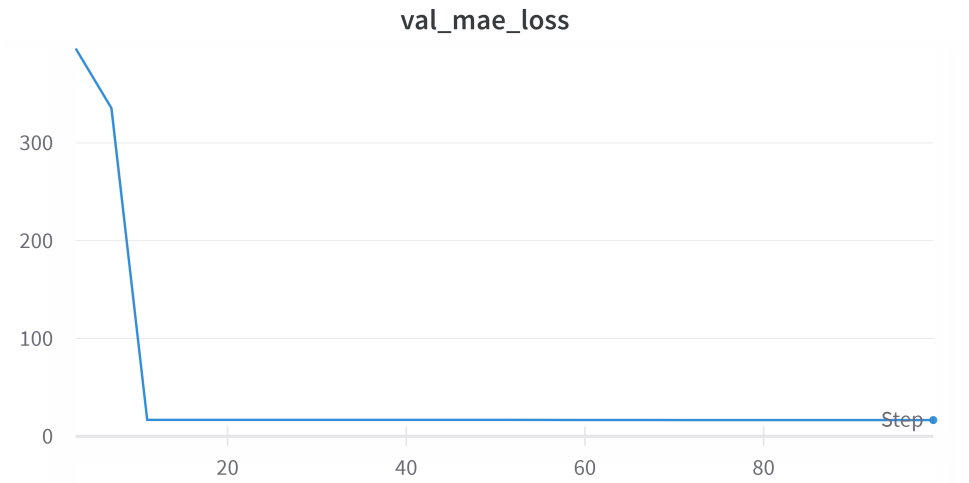


Figure 4.8: Validation MAE Loss (QT_interval) for model-A experiment with 5000 data sample

made it trigger learning rate decay to four times during the training process. The checkpoint reached epoch 42 where the validation MSE loss was 4167.65 and the validation MAE loss was 43.16. After that, a series of 25 unimproved epochs make the process early stop at epoch 67. The process of training illustrates through the journey of losses, as shown in figure 4.9 and figure 4.10.

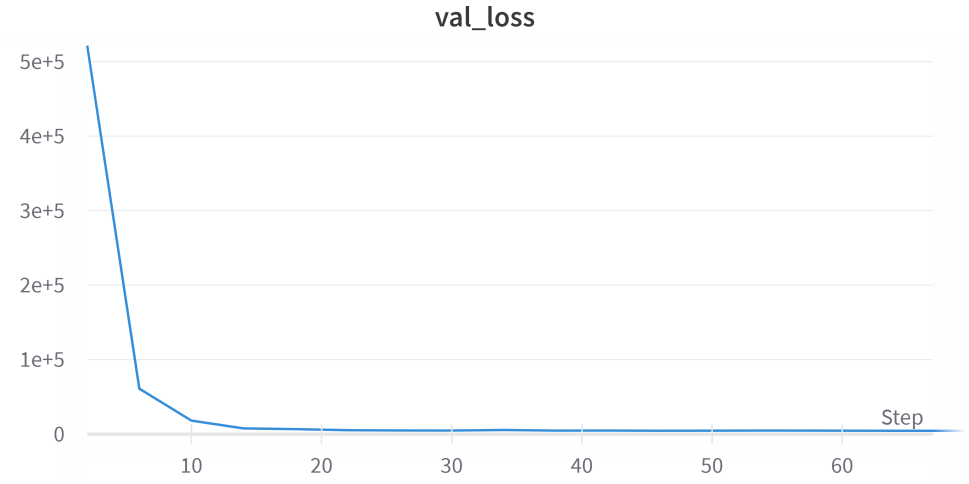


Figure 4.9: Validation MSE Loss (R_amplitude) for model-A training with 5000 data sample

Again validation errors of model-B training are compared against Zero-R estimate error (predicting the average). Table 4.6 provides the statistical details about the experiment 2, where model-A with encoder is trained with 5000 data samples of DeepFake dataset and predicting different features of ECGs one at a time. In the

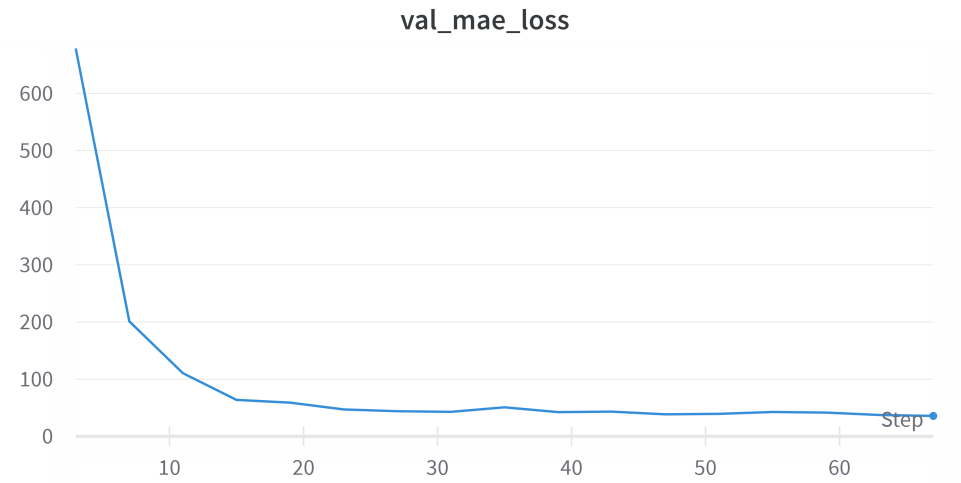


Figure 4.10: Validation MAE Loss (R_amplitude) for model-A training with 5000 data sample

| Variables | Validation error | | Zero-R error | |
|---|---|---|---|---|
| | MSE | MAE | MSE | MAE |
| Vent_rate (BPM) | 51.85 | 5.74 | 56.20 | 5.95 |
| QRS_duration (ms) | 69.15 | 6.89 | 74.04 | 6.88 |
| QT_interval (ms) | 298.58 | 16.56 | 418.42 | 16.38 |
| R_amplitude (µV) | 4167.65 | 43.162 | 55911.53 | 191.57 |

Table 4.6: Validation error and Zero-R error for model-A training on DeepFake ECG with training samples 5000.

table, we can see that model-A has almost beat the Zero-R error but it is tricky to say that it is not predicting average as values are almost near the average-predicting values.

## 4.6 Experiment 6 (Training of model-A with 13000-samples for QRS-duration, QT-interval and R-amplitude)

The results of experiment number 3 where model-A trained with 5000 show that the model has beat almost the Zero-R estimate error but it is very close to predicting the average. Since the training of model-A for predicting ventricular rate with 13000 showed a promising result in experiment 1, So another experimental setup is arranged where the prediction of QRS-duration, QT-interval, and R-amplitude with

three different training sessions of Model-A have done with 13000 training samples.

The rest of the configurations in the experimental setup for training are set similarly to table 4.3 and ground truth extraction are done similarly. The number of training data samples is set to 13000 from all normal DeepFake ECG with 90% training data and 10% validation data. Mean square error (MSE) is set as a loss function to the training and mean absolute error (MAE) is calculated for better interpretability of the loss journey. Following are the details for each training session:

- QRS_duration prediction's training had many no-improvement epochs in its training session. It did trigger three times the learning rate decays and finally trigger the early stopping at epoch number 37. The best checkpoint was obtained at the early phase at epoch number 13. The validation loss was 67.94 MSE and 6.62 MAE and figure 4.11 and 4.12 depicts losses journey during training.

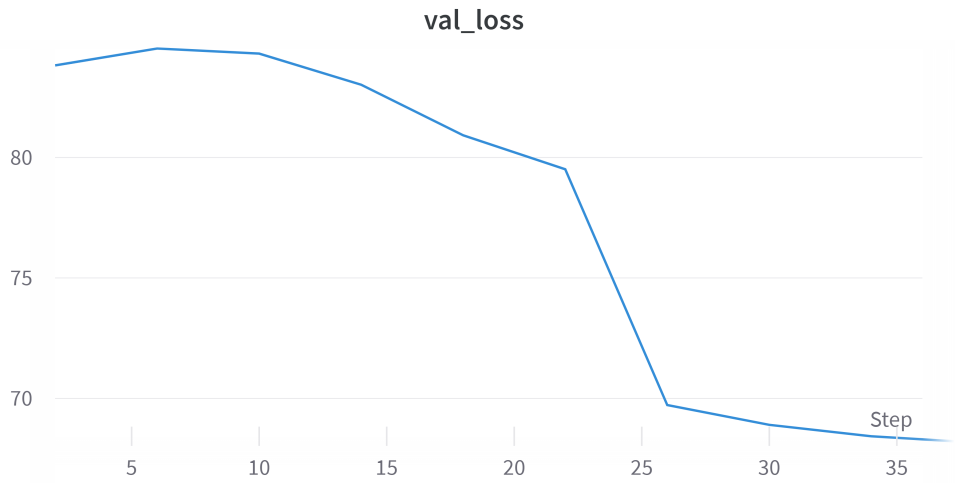


Figure 4.11: Validation MSE Loss (QRS_duration) for model-A training with 13000 data samples

- QT_interval 's training had several epochs with no improvement in the loss function. It only triggers two times the learning rates decay in the duration of *100* epochs. The best checkpoint was obtained at epoch *99* where the validation loss was *278.06* for MSE and *13.07* for MAE. The journey of validation MSE loss is shown in figure 4.13, while figure 4.14 illustrates MAE validation loss for QT_interval.

- R_amplitude feature prediction training on model-A has several epochs with no improvement. Total 5 times decay happened for learning rate but it did not trigger the early stopping and checkpoint obtained at epoch 78. The

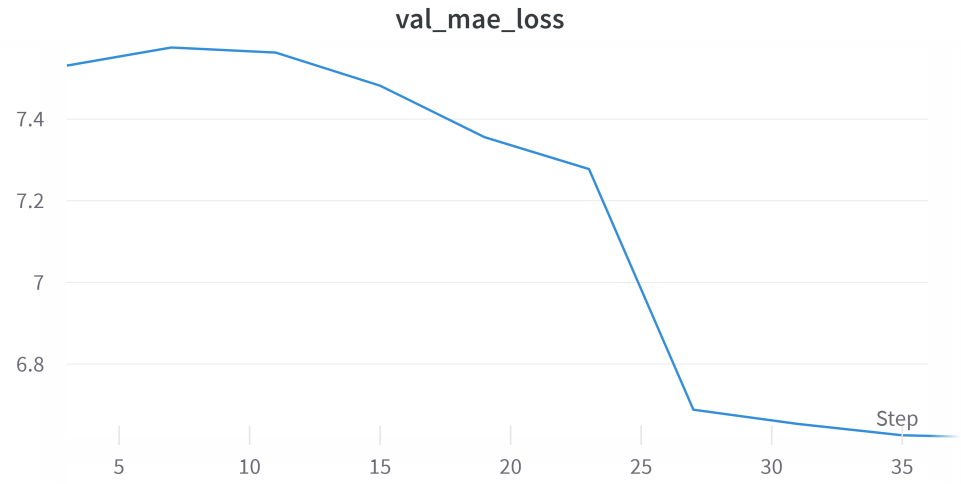


Figure 4.12: Validation MSE Loss (QRS_duration) for model-A training with 13000 data samples

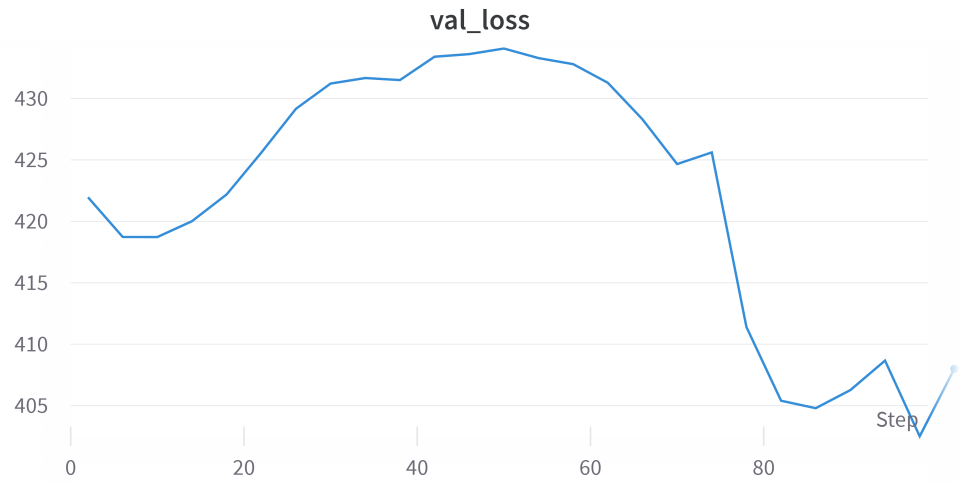


Figure 4.13: Validation MSE Loss (QT_interval) for model A training with 13000 data samples

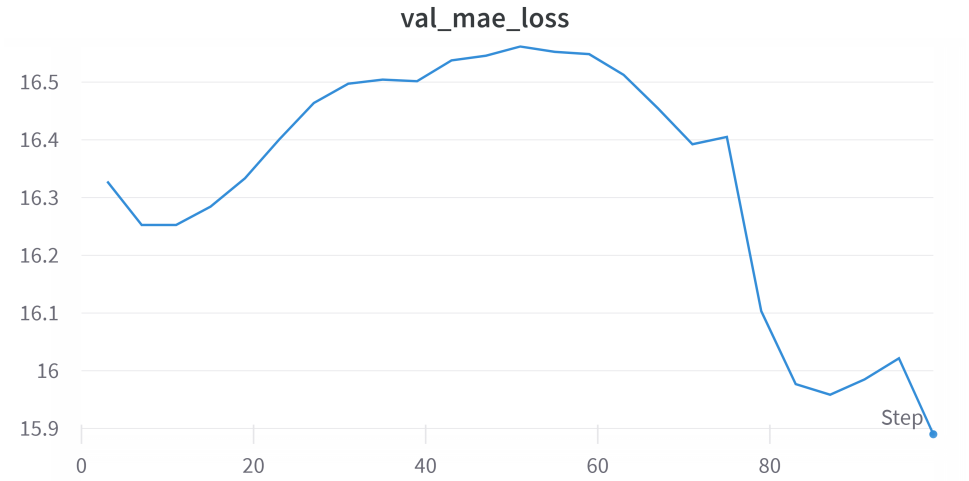


Figure 4.14: Validation MAE Loss (QT_interval) for model A training with 13000 data samples

validation loss at the epoch was 1872 MSE and 26.53 MAE. The whole journey for validation losses is shown in figure 4.15 and figure 4.16.

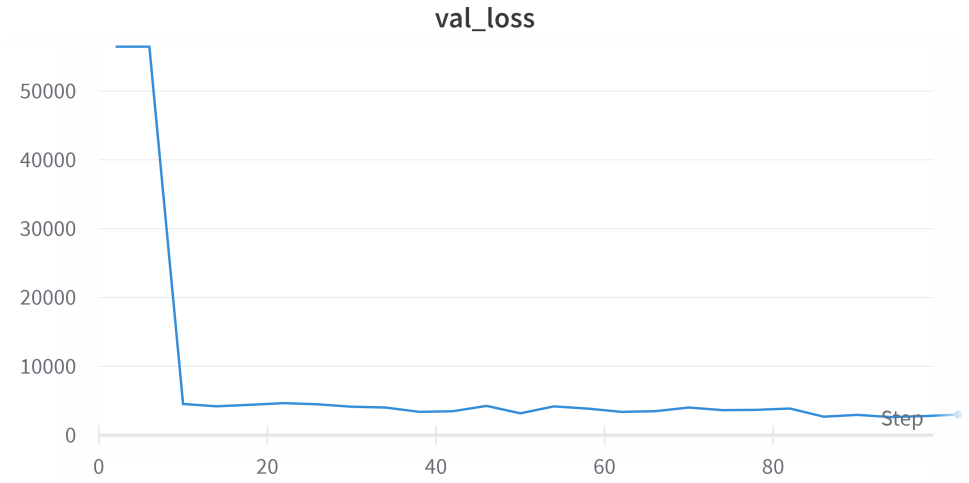


Figure 4.15: Validation MSE Loss (R_amplitude) for model-A training with 13000 data samples

At this stage, the losses of zero-R estimates were checked for 13000 training data samples of DeepFake ECGs and compared against this experiment training results and are shown in table 4.7. Model-A has shown as expected much better results with the 13000 samples from the dataset than the experiment before with 5000 data samples. Model-A has beat the zero-R estimates.

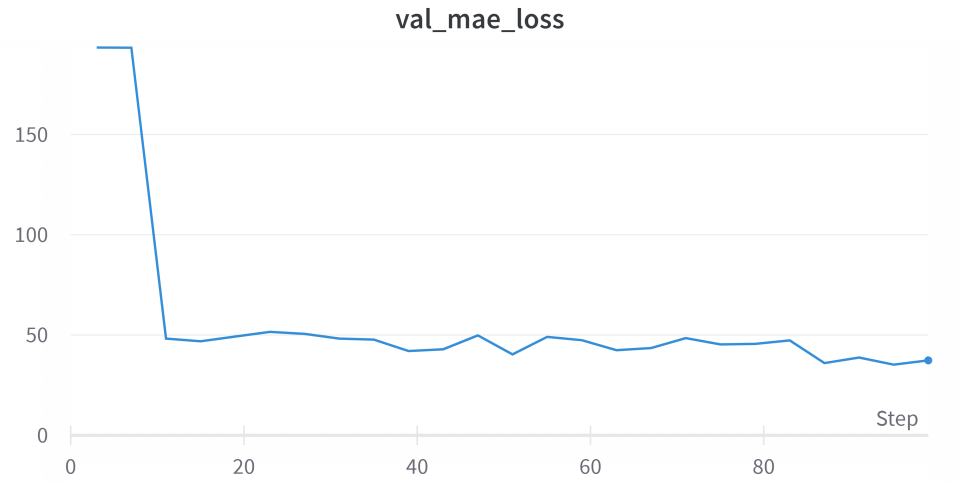


Figure 4.16: Validation MAE Loss (R_amplitude) for model-A training with 13000 data sample

| **Variables** | **Validation error** | | **Zero-R error** | |
|---|---|---|---|---|
| | MSE | MAE | MSE | MAE |
| Vent_rate (BPM) | 0.63 | - | 56.63 | 5.94 |
| QRS_duration (ms) | 67.94 | 6.62 | 74.72 | 6.89 |
| QT_interval (ms) | 278.06 | 13.07 | 419.06 | 16.36 |
| R_amplitude (µV) | 1873.08 | 26.53 | 56680.3 | 192.90 |

Table 4.7: Validation error and Zero-R error on DeepFake ECG for model-A training with 13000 datasamples.

## 4.7 Experiment 7 (Training of Model-B-Encoder-Decoder with limited data sample)

As model-B incorporated both the encoder and decoder of the transformer network, there are now more modules of powerful self-attention in comparison with model-A encoder-only. Therefore, a general expectation from model-B is to perform better. With this expectation, this work started training model-B with a limited data set i.e., 5000 samples from the DeepFake ECG dataset to best possibly utilize the computational resources available for this project. A total of four sessions of training were done for the model-B which predicted features like ventricular rate (vent_rate), QRS duration (QRS_duration), QT interval (QT_interval), and R peak amplitude (R_amplitude) from ECG signal. The configuration for parameters of the transformer network and training were kept the same as of model-A as shown in table 4.3. The extraction of lead-I ECGs was the same as model-A but the ground truth is extracted in a slightly different way where the shape of the ground truth has three dimensions. Splitting of data sample was similar to model-A with 90% training and 10% validation samples. MSE is kept training loss function and MAE is measured too for this experiment. Details about all four training sessions are briefly described below:

- Ventricular rate prediction training with a limited dataset had many no improvements in epochs during the session. It did activate the decay of the learning rate three times which ended the training session with early stopping at the stage of 33 epoch. The checkpoint was obtained at epoch 8 where the validation loss was 52.63 for MSE and 5.84 for MAE. The loss journey is shown in figure 4.17 and figure 4.18. The results were slightly opposite to the expectation of good results from model-B as compared to the result of the experiment from the same samples of model-A. Model-A for the prediction of ventricular rate has performed better than model-B.

- QRS duration prediction training started well in the beginning with a decrease of loss function but then no improved epochs stick with the training process. Total *8* learning rate decays were spotted during the session which finally ended the training session with early stopping at epoch *70*. The checkpoint was achieved at epoch *45* where MSE loss was *71.57* and MAE loss was *6.8*.the figure 4.19 and figure 4.20 show the loss graph. Again model-B for QRS duration did not beat model-A.

- QT interval has three learning rate decays in its training journey but with no early stopping. The checkpoint obtained at epoch *88* with *354.15* MSE validation

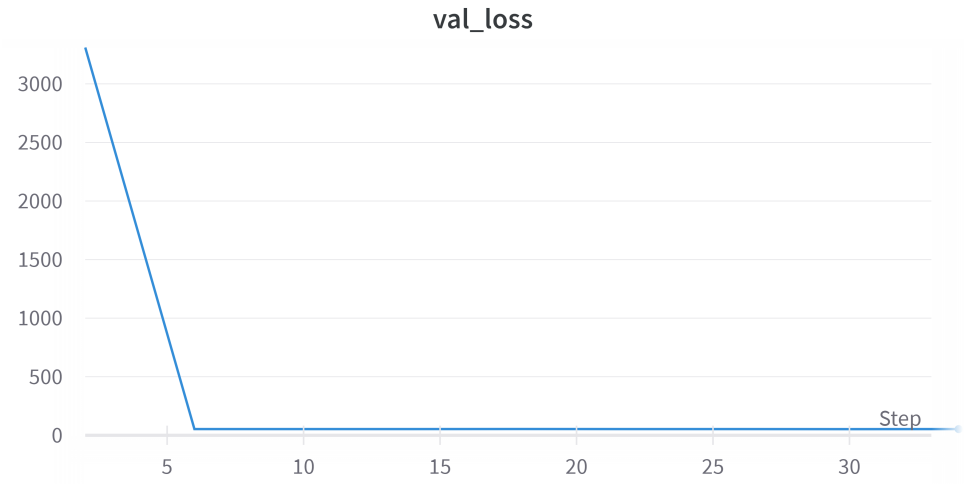


Figure 4.17: Validation MSE Loss (Vent_rate) of Model-B training with 5000 data sample


Figure 4.18: Validation MAE Loss (Vent_rate) of Model-B training with 5000 data sample

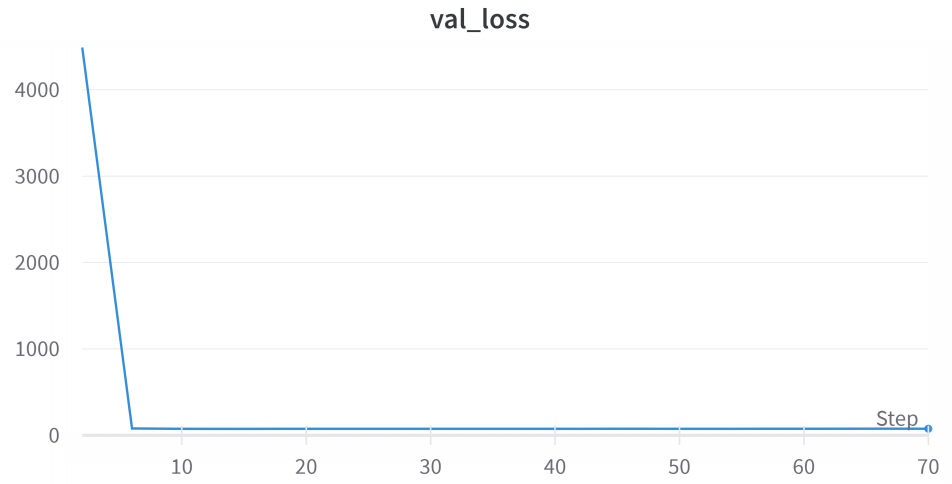


Figure 4.19: Validation MSE Loss (QRS_duration) of Model-B training with 5000 data sample

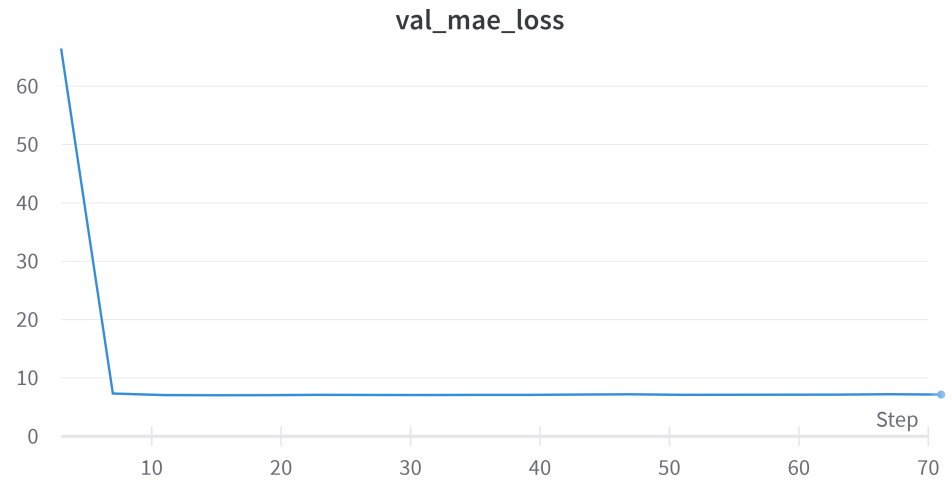


Figure 4.20: Validation MAE Loss (QRS_duration) of Model-B training with 5000 data sample

loss and *15.10* MAE validation loss. The losses are illustrated in figures 4.21 and 4.22. Again model-B for QT prediction did not beat model-A.

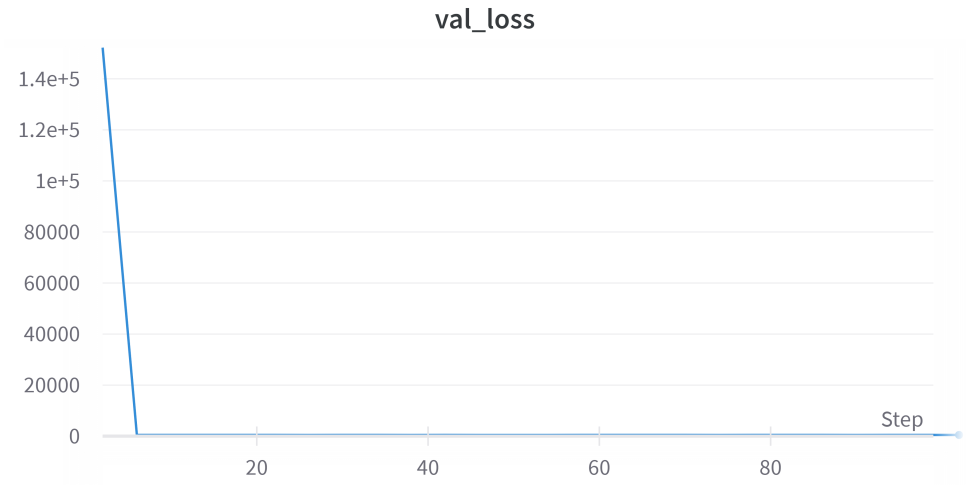


Figure 4.21: Validation MSE Loss (QT_interval) of Model-B training with 5000 data sample

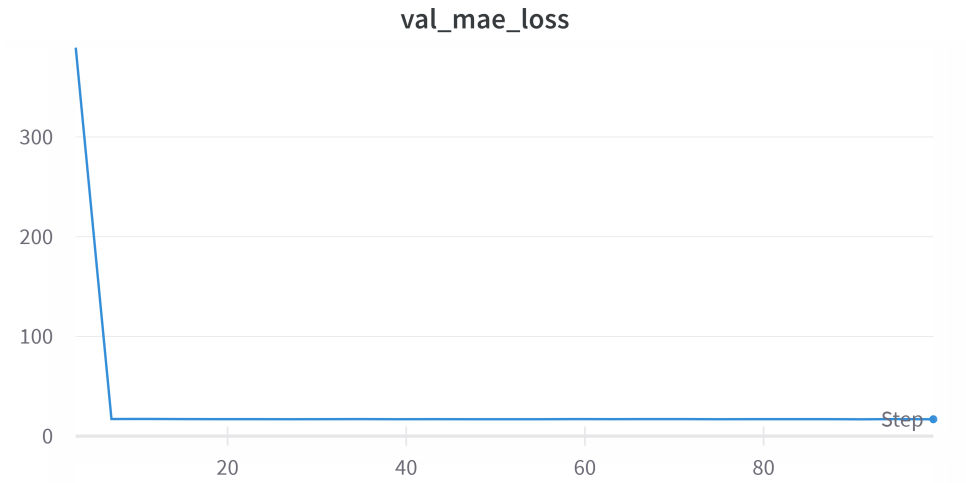


Figure 4.22: Validation MAE Loss (QT_interval) of Model-B training with 5000 data sample

- R peak amplitude prediction training has also three decays of learning rate and it did trigger the early stopping at epoch 49. the checkpoint achieved at epoch 24 where validation loss were 4151.58 MSE and 32.17 MAE. Figures 4.23 and 4.24 show the graph of the validation loss.

The results of Model-B are then compared with the zero-R errors score and it did beat the zero-R error in each feature category. But failed to beat the model A with similar 5000 data samples category.

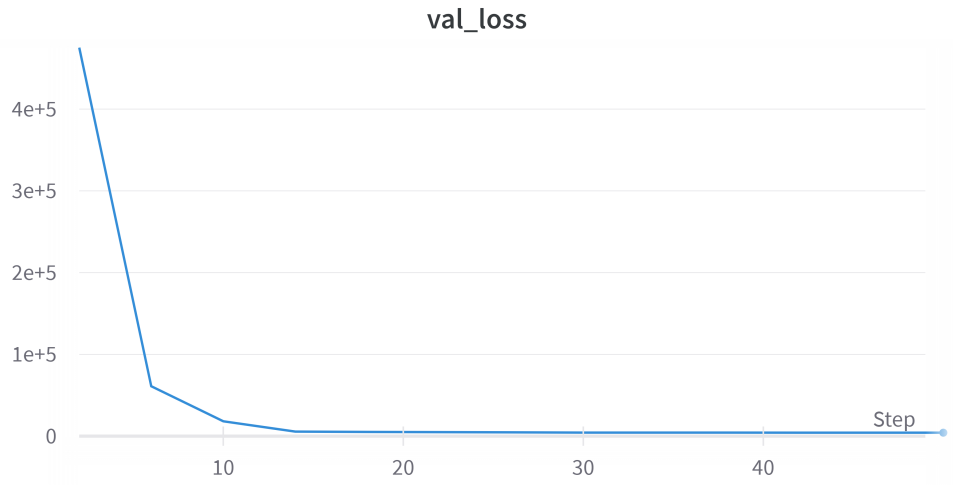


Figure 4.23: Validation MSE Loss (R_amplitude) of Model-B training with 5000 data sample

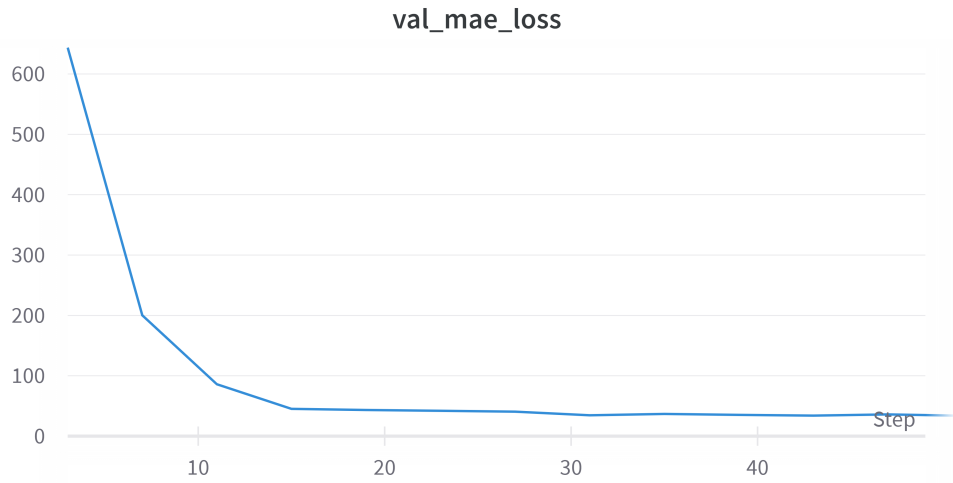


Figure 4.24: Validation MAE Loss (R_amplitude) of Model-B training with 5000 data sample

| **Variables** | **Validation error** | | **Zero-R error** | |
|---|---|---|---|---|
| | MSE | MAE | MSE | MAE |
| Vent_rate (BPM) | 52.63 | 5.84 | 56.20 | 5.95 |
| QRS_duration (ms) | 71.57 | 6.80 | 74.04 | 6.88 |
| QT_interval (ms) | 354.15 | 15.10 | 418.42 | 16.38 |
| R_amplitude (µV) | 4151.58 | 32.17 | 55911.53 | 191.57 |

Table 4.8: Validation error and Zero-R error on DeepFake ECG for model-B with training samples 5000.

## 4.8 Experiment 8 (Training Model-B with 13000 data-samples)

After the previous experiment where model-B did not show any better result than model-A, a chance to model-B given to training on large data samples like model-A was trained with *13000* data samples. All of the parameters for training and transformer network were kept the same as listed in table 4.3. The four sessions of training were done to predict ventricular rate, QRS duration, QT interval, and R peak amplitude. Mean square error (MSE) is the set training loss function for all sessions. The details of the training sessions are described below:

- Ventricular rate prediction training started well in the beginning with a decrease of loss function but then no improved epochs stick with the training process. A total of 3 learning rate decays were spotted during the session. The losses tended to increase and suddenly at the stage of the 32 epoch, it made the validation losses to the 'nan' value. Then later epochs gave a similar nan-validation loss. The training losses were continuously decreasing but validating losses were nan. Then forced a stop to train is done because it gives no point to train the model further. The checkpoint was achieved at epoch 7 where MSE loss was 49.66 and MAE loss was 5.4. Figure 4.25 and figure 4.26 show the loss graphs. Again model-B for ventricular rate did not beat model A for the 13000 data samples category.

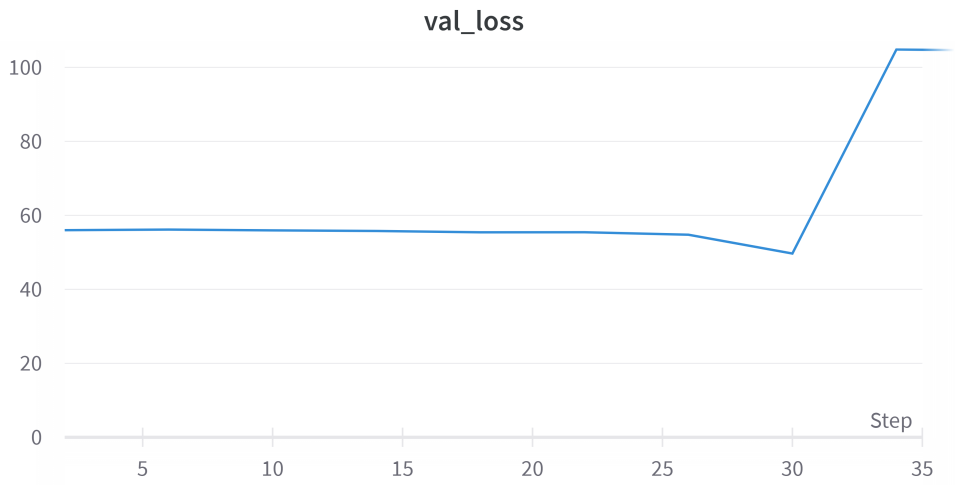


Figure 4.25: Validation MAE Loss (Vent_rate) of Model-B training with 13000 data sample

- The training session on QRS duration prediction was kept under tight observation. The training started well in the beginning with a decrease in loss

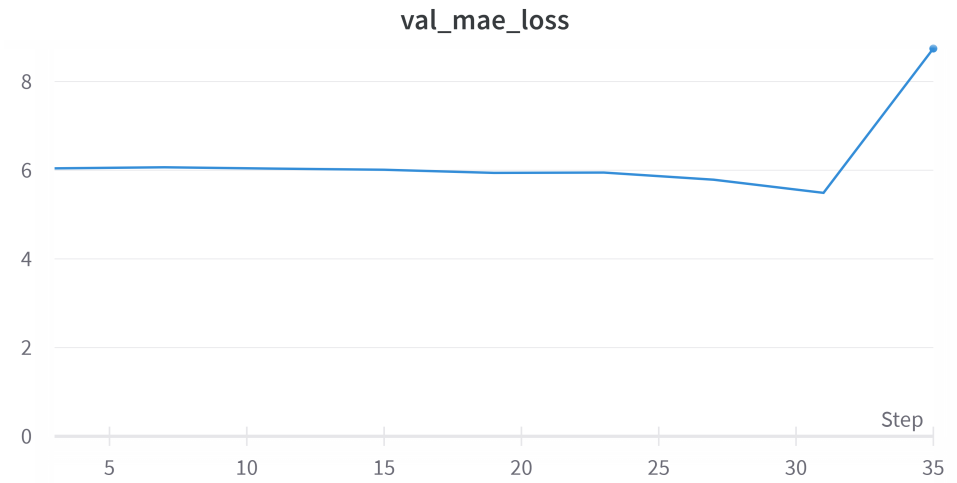


Figure 4.26: Validation MAE Loss (Vent_rate) of Model-B training with 13000 data sample

function but then some no improved epochs happens to the training process. Again losses started to increase and output the nan value for validation loss at 31 epoch. While the training loss continued to decrease. After obtaining the 'nan' value, the training was forced to stop to save the computation cost. The checkpoint was achieved at epoch 26 where MSE loss was 67.75 and MAE loss was 6.63. The figure 4.27 and figure 4.28 show the loss graph. Model-B loss functions were slightly little in comparison with model-A for QRS duration prediction.

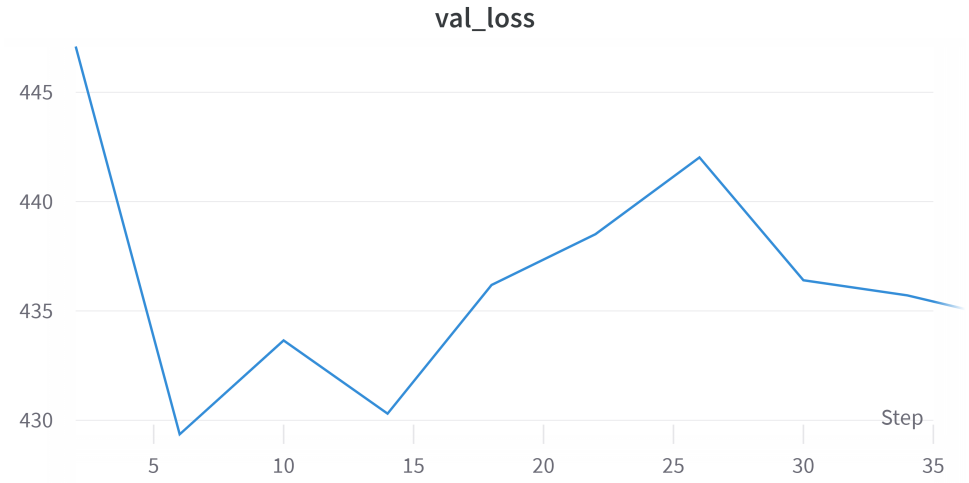


Figure 4.27: Validation MAE Loss (QRS_duration) of Model-B training with 13000 data sample

- QT interval prediction training shows a similar manner of spitting nan around

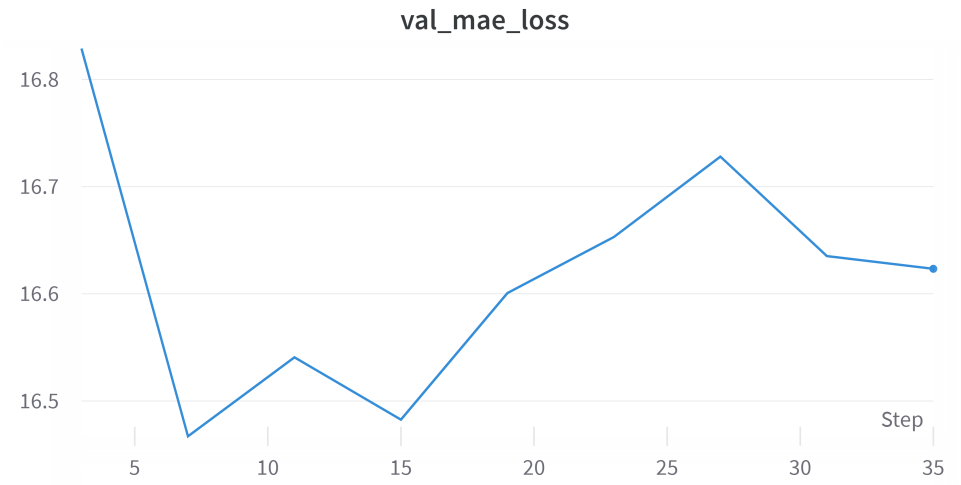


Figure 4.28: Validation MAE Loss (QRS_duration) of Model-B training with 13000 data sample

a similar stage of the session. This time at 30 epoch then forced stop was done to training. The checkpoint was achieved at epoch 29 where validation MSE loss and validation MAE loss were 406.66 and 16.12 respectively. Model-B results did not beat model-A for the prediction of QT interval from ECG. Figure 4.29 and figure 4.30 show the loss graphs for the validation.

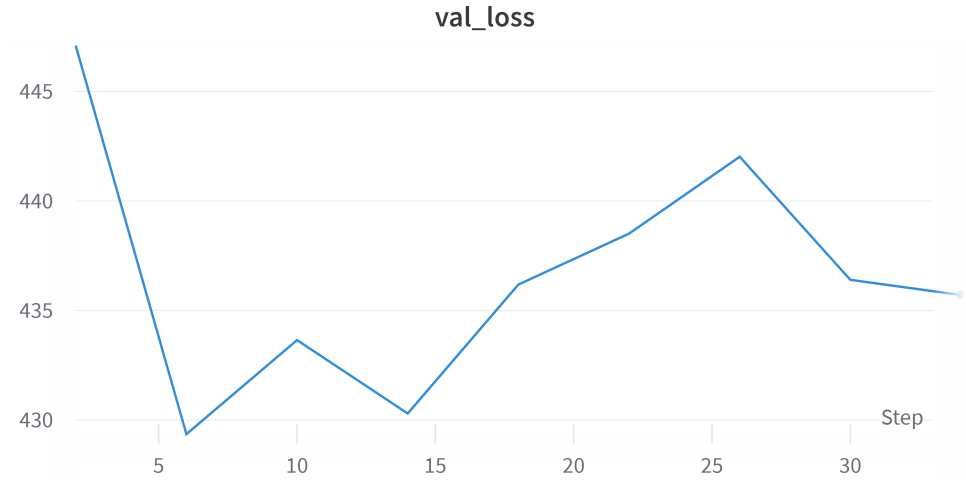


Figure 4.29: Validation MAE Loss (QT_interval) of Model-B training with 13000 data sample

- R peak amplitude prediction training shows a similar manner of where nan as loss output came around similar or bit earlier stage of the session. This time at *27* epoch then forced stop was done to training. The checkpoint was achieved at epoch *25* where validation MSE loss and validation MAE loss were *2570.31* and

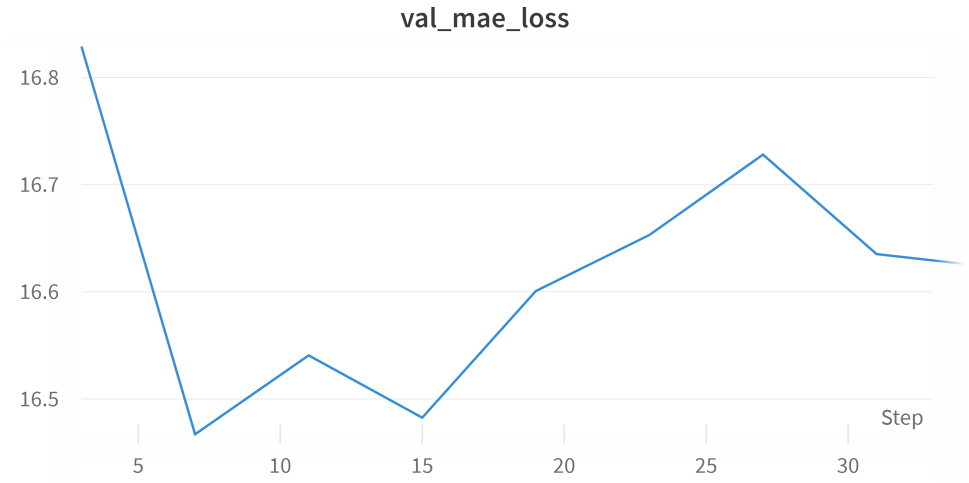


Figure 4.30: Validation MAE Loss (QT_interval) of Model-B training with 13000 data sample

*32.82* respectively. Model-B results did not beat model-A for the prediction of R amplitude from ECG.Figure 4.31 and figure 4.32 shows the loss graphs for the validation.

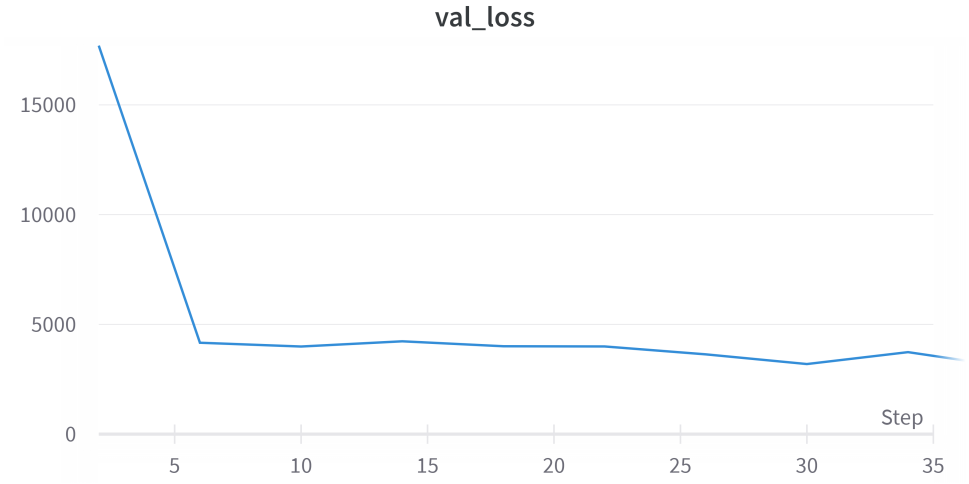


Figure 4.31: Validation MAE Loss (R_amplitude) of Model-B training with 13000 data sample

Though model-B did not beat model-A in terms of performance based on loss function it did beat the zero-R error (average estimating) for 13000 samples of the dataset for the training session. Comparison results are provided as shown in table 4.9. QT_interval started to give the nan value but since it already had 25 no improvement epochs so it stops due to early stopping and gave the final graph of the training and validating loss for the session as shown in figure 4.33. This graph

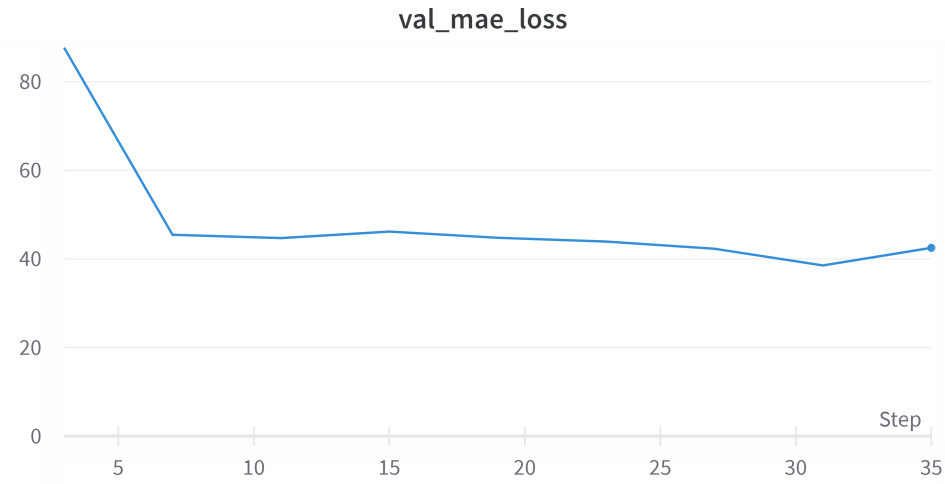


Figure 4.32: Validation MAE Loss (R_amplitude) of Model-B training with 13000 data sample

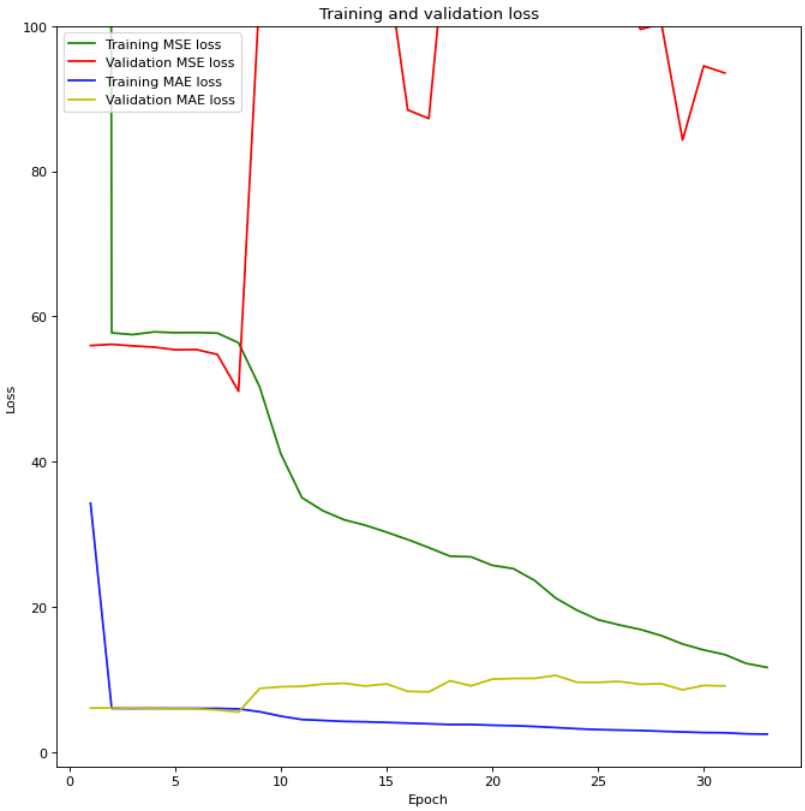


Figure 4.33: Graph illustrating the training and validation loss for (QT_interval) of Model-B training with 13000 data sample which shows overfitting of the model-B.

| Variables | Validation error | | Zero-R error | |
|---|---|---|---|---|
| | MSE | MAE | MSE | MAE |
| Vent_rate (ms) | 49.66 | 5.4 | 56.63 | 5.94 |
| QRS_duration (ms) | 67.75 | 6.63 | 74.72 | 6.89 |
| QT_interval (ms) | 406.66 | 16.12 | 419.06 | 16.36 |
| R_amplitude (µV) | 2570.31 | 32.82 | 56680.3 | 192.90 |

Table 4.9: Validation error and Zero-R error on DeepFake ECG for model-B training with 13000 data samples

| **Variables** | **Datasamples = 5000** | | | **Datasamples = 13000** | | |
|---|---|---|---|---|---|---|
| | **Model-A** | **Model-B** | **Zero-R** | **Model-A** | **Model-B** | **Zero-R** |
| **Ventricular rate** | 51.85 | 52.63 | 56.20 | 0.63 | 49.66 | 56.63 |
| **QRS_duration** | 69.15 | 71.57 | 74.04 | 67.94 | 67.75 | 74.72 |
| **QT_interval** | 298.58 | 354.15 | 418.42 | 278.06 | 406.66 | 419.06 |
| **R_amplitude** | 4167.65 | 4151.58 | 55911.55 | 1873.08 | 2570.31 | 56680.3 |

Table 4.10: Overall, MSE loss comparison between model-A, model-B and zero-R error for 5000 and 13000 datsamples

is showing the model is in the state of overfitting where training loss in green is decreasing while validating loss in red is increasing. The graph for the rest of the pieces of training was not recorded due to the forced stop of the training program after giving nan as validating loss to avoid the wastage of computational resources in Colab-pro.

## 4.9 Summary

To summarize the experiments of the project, this work first set the configuration parameters of the transformer network due to constraints of computational resources followed by the experiments for setting the training parameters like batch size, optimization parameter, etc. Later the training of both models for four parameters of ECG with 5000 data samples and 13000 data samples and compare the results against zero-R error. The summary graph for MSE loss and MAE loss for all training sessions are shown in table 4.10 and table 4.11 respectively. In addition the the zero-r error is also shown for 5000 and 13000 datasamples.

| Variables | Datasamples = 5000 | | | Datasamples = 13000 | | |
|---|---|---|---|---|---|---|
| | **Model-A** | **Model-B** | **Zero-R** | **Model-A** | **Model-B** | **Zero-R** |
| **Ventricular rate** | 5.74 | 5.84 | 5.95 | - | 5.4 | 5.94 |
| **QRS_duration** | 6.89 | 6.80 | 6.88 | 6.62 | 6.63 | 6.89 |
| **QT_interval** | 16.56 | 15.10 | 16.38 | 13.07 | 16.12 | 16.36 |
| **R_amplitude** | 43.162 | 32.17 | 191.57 | 26.53 | 32.82 | 192.90 |

Table 4.11: Overall, MAE loss comparison between model-A, model-B and zero-R error for 5000 and 13000 datsamples

# Chapter 5

# Discussion

This section will provide a detailed discussion of the findings of this research work. This will include a detailed description of the reasons behind the training results provided in chapter 4, the behavior of models during the training and possible factors that are affecting it, the role of components for designing of structure and configured parameters in models for it to learn analysis of ECG task.

This thesis work presented two models to evaluate the use of transformer networks for predicting the amplitudes and intervals from ECG signals. Both models did a reliable analysis of ECG and were able to predict the features of ECG like ventricular rate, QRS duration, QT interval, and R peak amplitude. Both models beat the zero-R estimate especially when training is done with a large number of data samples. After the detailed evaluation of the models' ability of ECG analysis and results provided in experiments in chapter 4 for validation primary metrics, it is visible that, model-A performs better than model-B for the task of ECG analysis.

Based on the results shown in chapter 4 of training sessions of both models, the embedded attentions from the encoder of the transformer networks of model-A did consist of the relevant information about the predicted feature and were accurate to perform the ECG analysis. However, the encoder-decoder structure of transformer networks in model-B was not able to beat the result of model-A (encoder only) regardless of further computations on embedded-attention from the encoder being done in the decoder part. So, in comparison to structure, ECG analysis using the encoder-only structure of transformer networks is less complex and required less computation and managed to outperformed the encoder-decoder transformer in the ECG analysis task. This also satisfies the reason behind why a majority of researchers [30], [28], [11], [14] developed the models using only the encoder of the transformer networks for different ECG tasks.

In transformer networks, the output from the encoder contains the same sequence length which was provided to its first encoder sub-layer, in this work

it was from the backbone component of the model i.e. embedded-encoded-ECG-signal. Similarly, the decoder output contains the same sequence length as the target source which provided its first decoder sub-layer which in this work was embedded-encoded-target-source. Both inputs were different from each other, i.e., one was embedded ECG sequence and another was embedded target sequence. Therefore, both outputs from the transformer network were different from each other. Encoder output was a long sequence of embedded attention while decoder output is a single sequence of embedded attention. The experiment for ablation of self-attention pooling layer 4.3 demonstrated that When keeping the designing of the structure of both models in mind, the sequence length of the output of either encoder or decoder played an important role, if its long sequence length then it needs transformation because prediction head is expecting a single sequence for the task of prediction, and if it is a single sequence (one) then no transformation is required. This transformation is called the self-attention pooling layer. Therefore, this self-attention pooling layer was only applied to Model-A where the output of the encoder was only used and required to be transformed before sending it to the predictor head of the model. Likewise, this self-attention was not applied in model-B as its output from the decoder is compatible with the prediction head, and no transformation was required.

Another important finding from the results is the over-fitting behavior of the encoder-decoder structure of model-B during the training with larger amount of data as shown in the figure 4.33 in chapter 4. It tended to increase the validation loss after a certain training period and after approximately 29 epochs session it started to give the 'nan' value as a validation loss. This is general behavior of model when it is overfitted. This behavior of transformer networks have heavily been questioned by the fellow researchers and developers on different deep learning community platform. The nature of 'nan' value output was different in many cases there but there was an almost similar story of 'nan' loss of the transformer networks which were queried by a user on "*stackoverflow*" platform [42], but it was not cleared from the query that the user was getting 'nan' value in training loss or in validating loss. If that was a training loss then this could be a different issue then what this thesis work have experienced i.e, overfitting in the transformer networks of model-B. However, a follow up solution were posted by same user which suggested the use of the higher configuration of the transformer networks' parameters which solved the issue of increased loss i.e, 'nan' value. Unfortunately, the lack of computational issues demonstrated in section 4.1 intervened the process of the evaluation of using the higher configuration of the transformer networks to avoid overfitting. Nevertheless, a discussion about the possible factors that tended model-B to overfit[48] are listed below :

- One of the reasons could be the over-complexity of the model-B. Since model-A did not show any overfitting behavior with the utilization of only the encoder of the transformer networks and model-B utilizes a similar type encoder along with the decoder. So, the problem could be happen due to the inclusion of the decoder of the transformer networks in the model-B, which made it complex to learn. The complexity of the model is an important consideration in deep learning, it plays an important role in learning the model because learning of the models depends upon the number of variables and features that a model needs to look into to make the predictions. Smaller networks might be easier to train but are not able to look into all information stored in data while bigger networks can look into different parameters but due to complexity end up in poor training and over-fitting. Model-B could have avoided attending to excessive information of parameters by using regularization techniques like dropout [39] in its transformer networks. This could have reduced the complexity by reducing the weight of the less important parameter in the model.

- Another reason for over-fitting could be the bad scheduling of early stopping [33] for model-B. Early stopping is a strategy to find the best stage of the training session where the model in neither in the under-fitting phase (i.e., the model is still learning) nor it is in the over-fitting phase. Therefore, the training of model-B should be stopped when the validation loss tended to increase.

- The strategy for splitting the dataset for training and validating the model-B used in this work is fixed and this could also be the reason for overfitting in model-B. Cross-validation technique [6] should be used in the splitting of data samples which allow the model to train and validate on different data sample to prevent overfitting.

The findings of this research work, however, are subjected to some limitations that intervene in the progress of the work. Some of the limitations of this work are discussed below:

- Lack of computational resources potentially limited the domain of experiments and evaluation which were done in this work. Firstly it intervened in some experiments to test the results with fully configured transformer networks which required high memory utilization from RAM.

- One of the major limitations this project faced is time constraints. Since this work is submitted as a short thesis to the educational institute, therefore, a short time duration i.e. almost four months is given as a timeline for the project.

During this duration intense work was done for literature research about ECG analysis and transformer networks, creating models for ECG analysis, dataset selection for evaluation purposes, implementations of models and training or validating routine, troubleshooting, and experiments for evaluation including training of models. Furthermore, the limitation of computational resources for training highly affects the time consumption of this work, especially, since Colab and its advanced version Colab-pro are inconsistent in efficiency in terms of time consumption. Sometimes Colab took three to four hours for training or sometimes eight or nine hours while data samples were the same. This heavy time consumption for the training of models constrained the domain of this research work. Hence a couple of trials for different ideas and methods were not done due to time constraints.

- As mentioned earlier in chapter 1 and chapter 2 of this report that there is a lack of availability of previous research where transformer networks were used for ECG analysis. In addition, the results of different works with their different methods for ECG analysis that applied to synthetic DeepFake datasets for ECG analysis were also not known. The prior research of the task usually helps to compare the results of the model. Therefore, this lack of information did constrain this project to compare the results of proposed models with transformer networks for ECG analysis using synthetic data with other methods available. However, this work considers this constraint as an opportunity to identify literature gaps in transformer networks and discover new research questions like the overfitting of transformer networks with larger data samples, etc. that open a new area of further study.

There were some research questions appeared during the designing phase of project time. These questions lead to many different methods and ideas that could have further improved the models. Scope of this project and the limitations faced by the project made to consider these research questions, ideas, and methods as future work for the transformer networks for ECG analysis. Some of the ideas are discussed below:

- The authors of the work [49] have argued that all *12* leads of ECG produced a better result in the deep-learning model than a single lead and for their work, the lead I, lead aVR, and lead V5 were best among the other leads. Based on their argument, the models of this work should also be trained with all 12 leads together, this will increase the load on computation resources and therefore be considered as future research work.

- The researchers have utilized segmented ECG signals in their work [19] and the

motivation behind this was the low computation complexities. Therefore, usage of segmented the ECG signal in this work may have lessened the input length of the sequence to the transformer and may have also reduced the computational load on resources. However, a research study is required to know the size of the segment of the signal which should contain the relevant information about the feature which the model is predicting. Therefore, the research of this method involves the thorough study of relevant features which make it out of the scope of this work, so, this work considers this method as future work to test the effectiveness of segmentation over the performance of transformer networks in ECG analysis.

- Since the model-B of this work showed a tendency of overfitting. One of the active areas of research for solving this issue could be the designated loss function for the task of ECG analysis involving transformer networks like the researchers have created Link constraints in their work [11]. This could also be done by researching the use of multiple layers of loss function within the model or including regularization techniques like [51], where dropouts techniques are developed for the transformer networks.

- It is important to understand which type of information the model is attending on from the ECG signal for deciding prediction [15]. Therefore, the visualization of embedded-attention output from the transformer network can give insights into the valuable parts of the ECG signal that plays a role in making decisions. This method of inclusion of visualization will make the model explanatory and reliable to doctors. This is another research area where the study is required to find out the best visualization methods for embedded- attention from the transformer networks and thus this work considers this as one of its future work.

- Ablation experiments are required to test the effect of the number of encoder and decoder layers, self-attention mechanism, positional encoding, feed-forward neural network layers of the transformer network, and the number of convolutional layers in the backbone on the performance of ECG analysis. This will provide a deep analysis and importance of the parts of the model.

- Another important research question obtained from the results and development of model-B is the research study of the designing of the number of queries as the target sequence to the decoder of model-B. This work believes that the design of sending ground truth like encoded-embedded-empty tensor, used in this work is not the quality input to the decoder. This topic requires thorough

research to find the best method for the creation of a target source for the decoder from an ECG signal.

In the reflection of the findings of this work, limitations faced during the research duration, and with the potential aims for future work, this work considers that the model-A is capable of predicting relevant features of the ECG signal which it is trained for. Though this AI-aided ECG analysis using transformer networks is in the early stage of implementation and several parameters are still required to be trained for analysis, still this work believes that the utility value of the project is integral to the motivation of this work of providing aid to doctors for accurate analyzation and better interpretations of ECG signal which leads to removing the failure of misdiagnoses of CVDs that usually happen when the doctor failed to analyze the ECG signal accurately. Model-A and its trained parameter can be integrated with any reliable ECG measuring machine. So ECG signals are measured with the help of electrodes and leads will display the different lead signals from different locations on the patient at the display machine. Before displaying the signal to the display, each signal is then tested against the model-A training for the prediction of the parameter ECG signal. Then displaying both the signal and parameters like amplitude and interval of the signal will provide the doctor with automatic annotation of the signal.

# Chapter 6

# Conclusion

This research study aimed to investigate the deep learning model of the transformer networks for the task of ECG analysis. The research work majorly focused on the two variants of the structure of transformer networks that was encoder-only and encoder-decoder for ECG analysis and investigations were done to find out which one is more effective and efficient in performing the ECG analysis. This work also wanted to address the existing gap of research questions for the transformer networks about the factors that affect the performance of the task while analyzing the ECG.

A thorough study was done for designing and developing the model for ECG analysis and implemented two end-to-end models i.e. model-A with encoder-only structure and model-B encoder-decoder structure of the transformer networks. Both models were given the chance to train up to 100 epochs. The Synthetic DeepFake ECG dataset with up to 13000 data samples was used for training models for different features of ECG like ventricular rate, QRS duration, QT interval, and R peak amplitude. Regression loss metric, mean squared error used to evaluate the performances of the model training. Based on their training results, both models were able to perform the ECG analysis and beat the Zero-R (estimating average) error.

Based on the quantitative analysis of the performance of models for ECG analysis, it can be concluded that model-A i.e. encoder-only variant of the structure of the transformer network performed better than the encoder-decoder variant for ECG analysis. This research has shown that the encoder-only variant is enough for performing ECG analysis and if combined with the decoder as done in model-B then it increased the complexity of the structure which resulted in overfitting. However, model-A is in its initial stage of training and only trained for a couple of parameters of the ECG signal, and further training is required for the rest of the parameters of the ECG, therefore, this work believes that after full training it can become a proper aid to doctors in performing ECG analysis.

This report discusses the several factors that play an important role in designing

the model like the self-attention pooling layer was only required for the encoder-only variant due to the sequence length of output from the encoder. Similarly, the creation of the target source is only required for the encoder-decoder variant of the transformer networks.

This report highlighted the potential methods that might improve the performance of ECG analysis i.e. instead of single-lead ECG signal input, the method for all 12 leads input should be designed for the transformer network. Furthermore, if the input ECG signal is segmented into small sequences then it might shed some computational load so that a fully configure transformer network could be evaluated for ECG analysis. Multiple layer loss and regularization techniques in transformer networks could help the encoder-decoder sequence to improve the results. In addition, thorough research is required to design the target source for the decoder for better ECG analysis. Therefor, further research to measure the outcomes of highlighted methods is needed in future work due to the limitations of this current work.

- Code availability

  The code used for developing, training, and validating the model is available on GitHub [13].

# Bibliography


[1] Jay Alammar. 'The Illustrated Transformer'. In: (27.06.2018). URL: https://jalammar.github.io/illustrated-transformer/.

[2] Roberto Andorno. In: 2.4 (2005), pp. 133–143. DOI: doi:10.1515/jibl.2005.2.4.133. URL: https://doi.org/10.1515/jibl.2005.2.4.133.

[3] Selcan Kaplan Berkaya, Alper Kursat Uysal, Efnan Sora Gunal, Semih Ergin, Serkan Gunal and M Bilginer Gulmezoglu. 'A survey on ECG analysis'. In: *Biomedical Signal Processing and Control* 43 (2018), pp. 216–235.

[4] Sebastian Bock and Martin Weiß. 'A Proof of Local Convergence for the Adam Optimizer'. In: *2019 International Joint Conference on Neural Networks (IJCNN)*. 2019, pp. 1–8. DOI: 10.1109/IJCNN.2019.8852239.

[5] DANIEL A BRODY. 'Ventricular rate patterns in atrial fibrillation'. In: *Circulation* 41.5 (1970), pp. 733–735.

[6] Michael W Browne. 'Cross-validation methods'. In: *Journal of mathematical psychology* 44.1 (2000), pp. 108–132.

[7] Pei-Xuan Cai, Yao-Chung Fan and Fang-Yie Leu. 'Compare Encoder-Decoder, Encoder-Only, and Decoder-Only Architectures for Text Generation on Low-Resource Datasets'. In: *Advances on Broad-Band Wireless Computing, Communication and Applications*. Ed. by Leonard Barolli. Cham: Springer International Publishing, 2022, pp. 216–225.

[8] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov and Sergey Zagoruyko. 'End-to-End Object Detection with Transformers'. In: *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I*. Glasgow, United Kingdom: Springer-Verlag, 2020, pp. 213–229. ISBN: 978-3-030-58451-1. DOI: 10.1007/978-3-030-58452-8_13. URL: https://doi.org/10.1007/978-3-030-58452-8_13.

[9] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov and Sergey Zagoruyko. 'End-to-end object detection with transformers'. In: *Computer Vision–ECCV 2020: 16th European Conference,*

*Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*. Springer. 2020, pp. 213–229.

[10] Joseph T Catalano. *Guide to ECG analysis*. Lippincott Williams & Wilkins, 2002.

[11] Chao Che, Peiliang Zhang, Min Zhu, Yue Qu and Bo Jin. 'Constrained transformer network for ECG signal processing and arrhythmia classification'. In: *BMC Medical Informatics and Decision Making* 21 (June 2021). DOI: 10.1186/s12911-021-01546-2.

[12] *DeepFake ECG*. URL: https://%20osf.%20io/%206hved/.

[13] *ECG_Analysis_using_Transformer_Network*. 2023. URL: https://github.com/SyedaAmbreenYawars345523/ECG_Analysis_using_Transformer_Network (visited on 13/05/2023).

[14] Bjørn Hansen. 'Heart Disease Classification using Transformer in Pytorch'. In: (26.07.2021). URL: https://towardsdatascience.com/heart-disease-classification-using-transformers-in-pytorch-8dbd277e079.

[15] Steven Hicks, Jonas Isaksen, Vajira Thambawita, Jonas Ghouse, Gustav Ahlberg, Allan Linneberg, Niels Grarup, Inga Strumke, Christina Ellervik, Morten Olesen, Torben Hansen, Claus Graff, Niels-Henrik Holstein-Rathlou, Pål Halvorsen, Mary Maleckar, Michael Riegler and Jorgen Kanters. 'Explaining deep neural networks for knowledge discovery in electrocardiogram analysis'. In: *Scientific Reports* (May 2021). DOI: 10.1038/s41598-021-90285-5.

[16] B F Hoffman and M R Rosen. 'Cellular mechanisms for cardiac arrhythmias.' In: *Circulation Research* 49.1 (1981), pp. 1–15. DOI: 10.1161/01.RES.49.1.1. eprint: https://www.ahajournals.org/doi/pdf/10.1161/01.RES.49.1.1. URL: https://www.ahajournals.org/doi/abs/10.1161/01.RES.49.1.1.

[17] Shenda Hong, Yuxi Zhou, Junyuan Shang, Cao Xiao and J. Sun. *Opportunities and Challenges in Deep Learning Methods on Electrocardiogram Data: A Systematic Review*. In: (Dec. 2019).

[18] Jie Hu, Li Shen and Gang Sun. 'Squeeze-and-excitation networks'. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 7132–7141.

[19] Rui Hu, Jie Chen and Li Zhou. 'A transformer-based deep neural network for arrhythmia detection using continuous ECG signals'. In: *Computers in Biology and Medicine* 144 (Feb. 2022), p. 105325. DOI: 10.1016/j.compbiomed.2022.105325.

[20] Aryan Jadon, Avinash Patil and Shruti Jadon. 'A Comprehensive Survey of Regression Based Loss Functions for Time Series Forecasting'. In: *arXiv preprint arXiv:2211.02989* (2022).

[21] Christian Rimer Juhl, IM Miller, GB Jemec, JK Kanters and C Ellervik. 'Hidradenitis suppurativa and electrocardiographic changes: a cross-sectional population study'. In: *British Journal of Dermatology* 178.1 (2018), pp. 222–228.

[22] Kadircan H. Keskinbora. 'Medical ethics considerations on artificial intelligence'. In: *Journal of Clinical Neuroscience* 64 (2019), pp. 277–282. ISSN: 0967-5868. DOI: https://doi.org/10.1016/j.jocn.2019.03.001. URL: https://www.sciencedirect.com/science/article/pii/S0967586819300256.

[23] Kirti Khunti. 'Accurate interpretation of the 12-lead ECG electrode placement: A systematic review'. In: *Health Education Journal* 73.5 (2014), pp. 610–623. DOI: 10.1177/0017896912472328. eprint: https://doi.org/10.1177/0017896912472328. URL: https://doi.org/10.1177/0017896912472328.

[24] Kikaben. 'Positional Encoding'. In: (31.10.2021). URL: https://kikaben.com/transformers-positional-encoding/.

[25] *LANGUAGE MODELING WITH NN.TRANSFORMER AND TORCHTEXT*. 2023. URL: https://pytorch.org/tutorials/beginner/transformer_tutorial.html (visited on 14/05/2023).

[26] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov and Luke Zettlemoyer. 'Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension'. In: *arXiv preprint arXiv:1910.13461* (2019).

[27] Yaowei Li, Yao Zhang, Lina Zhao, Yang Zhang, Chengyu Liu, Li Zhang, Liuxin Zhang, Zhensheng Li, Binhua Wang, EYK Ng et al. 'Combining convolutional neural network and distance distribution matrix for identification of congestive heart failure'. In: *IEEE Access* 6 (2018), pp. 39734–39744.

[28] Minghao Liu, Shengqi Ren, Siyuan Ma, Jiahui Jiao, Yizhou Chen, Zhiguang Wang and Wei Song. 'Gated Transformer Networks for Multivariate Time Series Classification'. In: *ArXiv* abs/2103.14438 (2021).

[29] Faidon Mitzalis, Ozan Caglayan, Pranava Madhyastha and Lucia Specia. 'BERTGEN: Multi-task Generation through BERT'. In: *arXiv preprint arXiv:2106.03484* (2021).

[30] Annamalai Natarajan, Yale Chang, Sara Mariani, Asif Rahman, Gregory Boverman, Shruti Gopal Vij and Jonathan Rubin. 'A Wide and Deep Transformer Neural Network for 12-Lead ECG Classification'. In: *2020 Computing in Cardiology* (2020), pp. 1–4.

[31] *PyTorch*. 2023. URL: https://www.nvidia.com/en-us/glossary/data-science/pytorch/ (visited on 13/05/2023).

[32] Jun Qi, Jun Du, Sabato Marco Siniscalchi, Xiaoli Ma and Chin-Hui Lee. 'On mean absolute error for deep neural network based vector-to-vector regression'. In: *IEEE Signal Processing Letters* 27 (2020), pp. 1485–1489.

[33] Garvesh Raskutti, Martin J Wainwright and Bin Yu. 'Early stopping and non-parametric regression: an optimal data-dependent stopping rule'. In: *The Journal of Machine Learning Research* 15.1 (2014), pp. 335–366.

[34] Pooyan Safari, Miquel India and Javier Hernando. 'Self-Attention Encoding and Pooling for Speaker Recognition'. In: Oct. 2020, pp. 941–945. DOI: 10.21437/Interspeech.2020-1446.

[35] Sanjay Tanaji Sanamdikar, Dr. S. T. Hamde and Dr. V. G. Asutkar. 'A Literature Review on Arrhythmia Analysis of ECG Signal'. In: 2015.

[36] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov and Liang-Chieh Chen. 'Mobilenetv2: Inverted residuals and linear bottlenecks'. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 4510–4520.

[37] Patrick Schwab, Gaetano C Scebba, Jia Zhang, Marco Delai and Walter Karlen. 'Beat by beat: Classifying cardiac arrhythmias with recurrent neural networks'. In: *2017 Computing in Cardiology (CinC)*. IEEE. 2017, pp. 1–4.

[38] RR Selvaraju, M Cogswell, A Das, R Vedantam, D Parikh and D Batra. 'Grad-CAM: Visual explanations from deep networks via gradient-based localization. arXiv 2016'. In: *arXiv preprint arXiv:1610.02391* ().

[39] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever and Ruslan Salakhutdinov. 'Dropout: a simple way to prevent neural networks from overfitting'. In: *The journal of machine learning research* 15.1 (2014), pp. 1929–1958.

[40] Vajira Lasantha Thambawita, Jonas L Isaksen, Steven Hicks, Jonas Ghouse, Gustav Ahlberg, Allan Linneberg, Niels Grarup, Christina Ellervik, Morten Salling Olesen, Torben Hansen, Claus Graff, Niels-Henrik Holstein-Rathlou, Inga Strümke, Hugo L. Hammer, Mary M Maleckar, Pål Halvorsen, Michael A. Riegler and Jørgen K. Kanters. 'DeepFake electrocardiograms: the key for open

science for artificial intelligence in medicine'. In: *medRxiv* (2021). DOI: 10.1101/2021.04.27.21256189. URL: https://doi.org/10.1101/2021.04.27.21256189.

[41] Kristian Thygesen, Joseph S Alpert, Harvey D White, TASK FORCE MEMBERS: Chairpersons: Kristian Thygesen (Denmark) et al. 'Universal definition of myocardial infarction'. In: *circulation* 116.22 (2007), pp. 2634–2653.

[42] *Transformer Model Output Nan Values in Pytorch*. 2021. URL: https://stackoverflow.com/questions/66542007/transformer-model-output-nan-values-in-pytorch (visited on 10/04/2022).

[43] *Unipolar and Bipolar connections*. 2015. URL: https://www.wikilectures.eu/w/Unipolar_and_bipolar_connection (visited on 14/05/2023).

[44] Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser and Illia Polosukhin. 'Attention is All you Need'. In: *ArXiv* abs/1706.03762 (2017).

[45] *Welcom to Colab*. 2023. URL: https://colab.research.google.com/ (visited on 10/05/2020).

[46] Qingsong Wen, Tian Zhou, Chaoli Zhang, Weiqi Chen, Ziqing Ma, Junchi Yan and Liang Sun. 'Transformers in Time Series: A Survey'. In: (Feb. 2022).

[47] *World Health Organisation, Cardiovascular Diseases (CVDs)*. 2021. URL: https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds) (visited on 02/05/2023).

[48] Xue Ying. 'An overview of overfitting and its solutions'. In: *Journal of physics: Conference series*. Vol. 1168. IOP Publishing. 2019, p. 022022.

[49] Dongdong Zhang, Samuel Yang, Xiaohui Yuan and Ping Zhang. 'Interpretable deep learning for automatic diagnosis of 12-lead electrocardiogram'. In: *Iscience* 24.4 (2021), p. 102373.

[50] Zhiqiang Zhao, Ziheng Jia and Tong Liu. 'Ominous T-Wave Changes in an Older Adult With Chest Pain'. In: *JAMA Internal Medicine* 183.1 (2023), pp. 78–79.

[51] Wangchunshu Zhou, Tao Ge, Ke Xu, Furu Wei and Ming Zhou. 'Scheduled drophead: A regularization method for transformer models'. In: *arXiv preprint arXiv:2004.13342* (2020).

[52] Zhenxun Zhuang, Mingrui Liu, Ashok Cutkosky and Francesco Orabona. 'Understanding adamw through proximal methods and scale-freeness'. In: *arXiv preprint arXiv:2202.00089* (2022).