Master Thesis

Single-lead ECG classification based on Transformer models

Kilian Laurin Kramer

Thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science of Artificial Intelligence
at the Department of Advanced Computing Sciences
of the Maastricht University

Thesis Committee:

Prof Dr. Stef Zeemering
Prof Dr. Pietro Bonizzi
Prof. Dr. Joël Karel

Maastricht University
Faculty of Science and Engineering
Department of Advanced Computing Sciences

September 22, 2023

Contents

Abstract

The accurate classification of electrocardiogram (ECG) signals is crucial for diagnosing various cardiovascular diseases, such as Atrial Fibrillation and to provide cardiologists with reliable predictions. With recent advances in deep learning, Transformer models [1] have emerged as powerful tools for the accurate single and multi-lead ECG classification of Atrial Fibrillation [3] [5] [9] [18] [20] [22] [36] [39]. This thesis investigates the potential of applying Transformer-based models to single-lead ECG classification and provides a comparison with non deep learning (based on extracted features) and other deep learning based models, such as Convolutional Networks.

Chapter 1

Introduction

With recent advances in deep learning models, Transformer-based architectures have emerged as powerful tools for accurate single-lead and multi-lead ECG classification [3] [5] [9] [20] [18] [22] [36] [39]. Multi-lead ECGs provide a view of the heart's electrical activity from different angles and can localise abnormalities more reliably. It is a standard in clinical monitoring. Accurate single-lead ECG classification is an attractive area of research when it comes to continuous processing of single-lead ECGs by remote monitoring systems, such as wearable devices like Apple Watches [18], for continuous monitoring of an individual's condition to provide early suggestions for a doctor's visit. This thesis investigates the potential of applying Transformer-based models to ECG classification and provides a comparison with non deep learning (based on extracted features) and other deep learning based models, such as Convolutional Networks. The experiments focus in particular on the classification of Sinus Rhythm, Atrial Fibrillation, Atrial Flutter, Premature Atrial Contractions and Premature Ventricular Contractions.

1.1 Problem statement

1.1.1 Electrocardiograms

Electrocardiograms (ECGs) monitor the condition of a patient's heart. An ECG measures the electricity flowing through the heart in repeated cardiac cycles. ECGs are an essential tool for cardiologists to diagnose heart diseases. To measure the heart's activity, electrodes (called leads within the ECG) are placed on the skin of the upper chest and back. The number of leads affects the quality of the measurements, i.e. single-lead ECGs are based on two electrodes, while multi-lead ECGs use more than two electrodes. Multi-lead ECGs can contain up to 12 leads, defined in the literature as I, II, III, aVR, aVL, aVF, V1, V2, V3, V4, V5 and V6 [2] [21]. Multi-lead ECGs are used in preference to single-lead ECGs because they provide a view of the heart from different angles and can therefore localise abnormalities more accurate through spatiality. Typically, these recordings last from a few seconds to a few minutes and are called short-term ECGs. However, sometimes long-term ECGs are performed. Long-term ECGs monitor the heart's activity over a longer period of time, e.g. 24 hours, and provide a more complete picture. This is necessary to detect abnormalities that are less common and harder to detect. The scope of this thesis focuses on short-term single-lead ECGs. An electrocardiogram can be recorded with different devices, e.g. in clinical settings 12-lead recordings are standard, while for remote monitoring often portable devices are used, such as the Holter monitoring system. The Holter monitoring system can be adapted to different numbers of leads as required, e.g. three leads or up to 12

leads. Special watches [18] can now also be used to record the heart's activity. These watches use a technology called Photoplethysmography (PPG), where LED lights under the watch emit light into the skin [26]. The light is absorbed and reflected by blood vessels and the changes in light absorption are measured to determine heart rate. However, PPG is a measure of the blood pressure from which the heart rate can be derived, but which is not direct an ECG recording of the electrical activity of the heart. Usually, the derived quality of the sensor is not as accurate as a standard ECG, making it a less reliable method. A watch is still an interesting device because it can monitor a patient's heart condition on a regular basis. It also has the advantage of not requiring the involvement of a cardiologist and can provide possible advice for a more transparent in-house check in a doctor's surgery. Accuracy in arrhythmia detection is important for several reasons, including arrhythmia risk and correct treatment. For example, increased accuracy in automated ECG processing can reduce false positives and false negatives. This can avoid unnecessary doctoral visits and lead to more efficient and effective clinical monitoring and treatment. In addition to developing accurate models, a goal of this thesis will be to transfer and apply the pre-trained models from the Physionet 2021 challenge data [28] to the database provided by the University of Maastricht, here referred to as the MyDiagnostick database. Both datasets contain recordings from different monitoring systems. Analysis and pre-processing steps need to take into account the type of monitoring system and electrodes, number of leads, recording duration, sampling rate and post-processed filters applied, which will be discussed in section 4.1.3.

Figures 1.1 and 1.2 illustrate the activity of a heart from a physiological perspective and a single normal heartbeat from an ECG, defined as a Sinus Rhythm. In the literature [2] [21] a heartbeat is described by the P, Q, R, S, T waves, segments and their intervals. Each wave and segment corresponds to a specific event in the electrical cycle of the heart. Below is a brief description of the entire contraction and depolarisation process of a single normal Sinus Rhythm heartbeat.

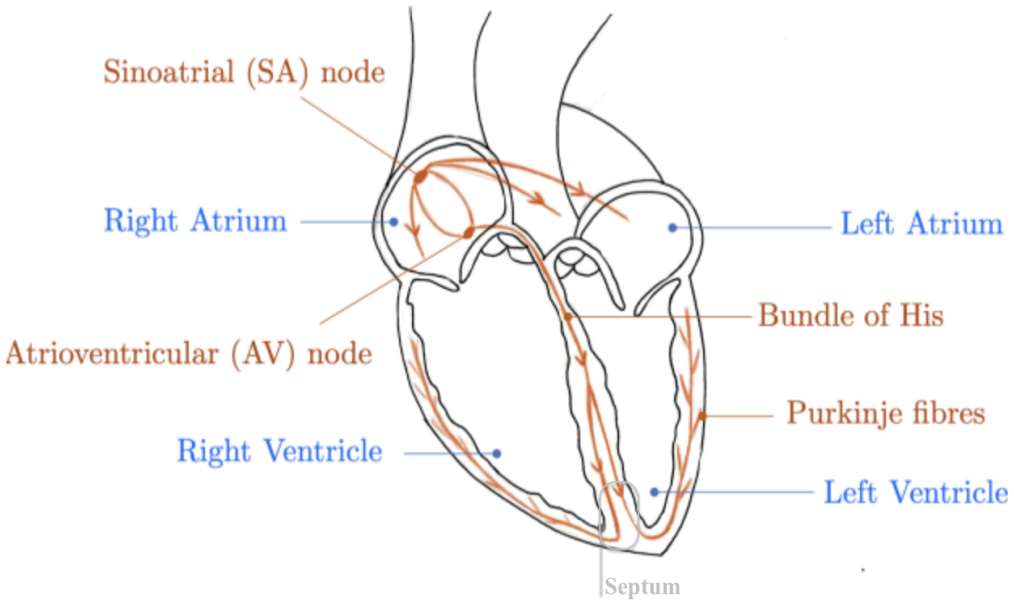

Figure 1.1: Human heart anatomy
Source: Collimator

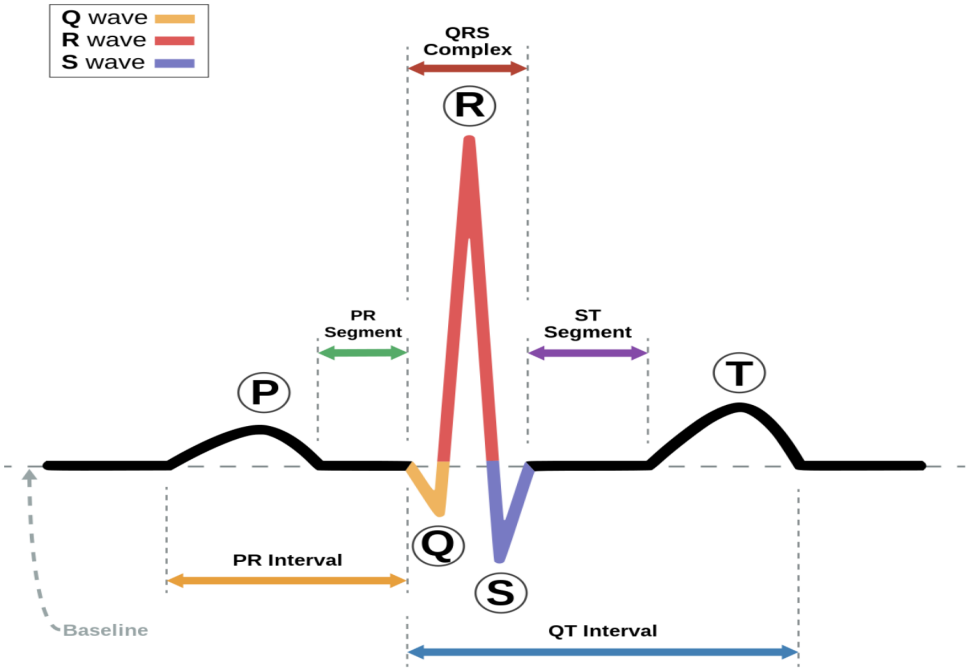

Figure 1.2: Heartbeat rhythm composition
Source: Wikipedia

The two graphs show a single heart contraction and depolarisation recorded on an ECG, starting with the electrical impulse from the Sinoatrial (SA) node and the atrial contraction, followed by the depolarisation phase and the pumping of blood from the ventricles into the aorta and pulmonary artery and the final repolarisation and relaxation of the heart. The SA node, located in the right atrium, is the heart's natural pacemaker. It initiates all heartbeats and determines the heart rate. The P wave in an ECG shows the electrical activation of the SA node, which causes the atria of the heart to contract. It is the first wave in a heartbeat and is usually small and positive. When the atria of the heart are filled with blood, the SA node fires and the electrical impulse from the SA node spreads to the two upper atria, causing them to contract. Atrial depolarisation, which fills the blood from the atria into the ventricles, begins about 100 milliseconds after the SA node fires. The Atrioventricular (AV) node, located opposite the SA node in the right atrium, receives the electrical signal from the SA node and marks the start of ventricular depolarisation. The electrical depolarisation is slowed down before propagating to the ventricles, here part of the PR interval. The PR segment represents the time it takes for the signal to travel from the SA node to the AV node. The AV node acts as an electrical gateway to the ventricles and delays the electrical conduction to the ventricles. This delay ensures that the atria contract all the blood into the ventricles before the ventricles depolarise. The AV node conducts the electrical impulse to the AV bundle (also called the Bundle of His). The bundle is divided into right and left branches, through which the electrical impulse is conducted to the Purkinje Fibres and causes the main depolarisation effect of the ventricles. This ventricular depolarisation is seen on the ECG as the QRS complex. The Q wave corresponds to the depolarisation of the interventricular septum. The R wave is produced by the depolarisation of the main mass of the ventricles. The S wave represents the final phase of ventricular depolarisation. Atrial repolarisation also occurs during this time, but the signal is obscured by the large QRS

complex. The ST segment, which follows the QRS complex, is the initial phase of the ventricular repolarisation. The ST segment reflects when the ventricles contract, pumping oxygen rich blood and oxygen poor blood into the aorta and pulmonary artery. The final T wave represents ventricular repolarisation before the heart relaxes. Repolarisation continues until the end of the T wave. The QT interval represents the total ventricular contraction activity (depolarisation and repolarisation). At the end of this process, the atria are filled with blood again and the SA node fires to repeat the cardiac cycle. This process represents a normal Sinus Rhythm.

1.1.2 Arrhythmia diseases

In the literature [2] [21], normal Sinus Rhythm is described as being within normal limits and ranges. For Sinus Rhythm, this is usually between 60 and 100 beats per minute. A rate below 60 is generally defined as Sinus Bradycardia and a rate above 100 beats per minute as Sinus Tachycardia. The two classes group several subtypes of arrhythmia. Arrhythmia subtypes are characterised by different origins and causes in the heart and need to be treated individually. In the Physionet 2021 challenge data, more than 100 arrhythmia subtypes are present 4.1. For the evaluation of the models transferred from the pre-training on the Physionet 2021 challenge data to the MyDiagnostick data, this thesis focuses the classification on Sinus Rhythm (SR), Atrial Fibrillation (AF), Atrial Flutter (AFL), Premature Atrial Contractions (PAC) and Premature Ventricular Contractions (PVC). This is due to the limited class annotation of the MyDiagnostick dataset, which only consists of these class labels. However, a broader performance evaluation of the models is performed on 26 different classes of the annotated Physionet 2021 challenge data. Most of the more than 100 available classes in the Physionet 2021 challenge data contain few examples, so the challenge uses a subset of 26 classes for the official scored metrics. Much research has been done to develop accurate models for binary classification of non-abnormality vs. abnormality, i.e. Sinus Rhythm vs. Atrial Fibrillation. However, fewer research has been done on distinguishing specific arrhythmia subtypes, such as Atrial Fibrillation and Atrial Flutter. The limited research in this area tends to show weaker outcomes [30] [29]. Atrial Fibrillation and Atrial Flutter are often confused by algorithms because they have similar characteristics. PAC and PVC are also examined because the MyDiagnostick dataset provided contains these annotations. In the following is a brief summary of each rhythm and arrhythmia type studied:

- Sinus Rhythm (SR): Regular P waves followed by a narrow QRS complex, typically lasting between 80 and 100 milliseconds. The PR interval remains constant throughout. Heartbeats are regular, between 60 and 100 beats per minute.

- Atrial Fibrillation (AF): Atrial fibrillation is abnormal electrical activity that causes the atrial muscle fibres to contract at different times. Atrial Fibrillation is characterised by a rapid and irregular heartbeat. These uncoordinated contractions produce a quivering or fibrillating activity. Atrial Fibrillation does not have constant P waves preceding the QRS complexes, although the fibrillation effect may resemble a P wave at times when it is not expected. Only some of the electrical signals are conducted down into the ventricles, resulting in ventricular depolarisation. However, there is no real pattern to which impulses are conducted. In the literature [2] [21], AF is also described as an irregular heart rhythm, which explains the variable lengths of the RR intervals.

- Atrial Flutter (AFL): Atrial Flutter is often confused with Atrial Fibrillation because of its similar characteristics. The main difference is that Atrial Flutter is characterised by coordinated electrical activity in the atria due to a re-entry pathway, resulting in rapid contraction of the atria. This is usually around 250 and 300 beats per minute with a regular

atrial rate and a narrow QRS complex. The AV conducts the signal slower, resulting into a slowed ventricular depolarization. Atrial Flutter is therefore characterised by the number of P waves compared to the number of ventricular contractions, which shows the ratio of non conducted to conducted beats, e.g. a 3:1 conduction means that every third atrial impulse is conducted to the ventricles. Atrial Flutter is often characterised by a "sawtooth" pattern of atrial activity, known as flutter waves, caused by the rapid atrial depolarisation. A conduction ratio of one to one is also possible and is associated with instability and progression to ventricular fibrillation. These ratios can be variable in the same patient, making the ventricular rate irregular, which is why it can often be mistaken for Atrial Fibrillation. Atrial Flutter is less dangerous than Atrial Fibrillation but can progress if left untreated.

- Premature Atrial Contractions (PAC): Premature Atrial Contractions are heartbeats that originate in the atria. PACs occur as early and extra beats that disrupt the regular heart rhythm. On an ECG, PAC shows a premature and often abnormal P wave followed by a QRS complex. Premature Atrial Contractions are characterised by a different P wave morphology compared to the normal sinus P wave, which follows a narrow QRS complex. The beat following the PAC may resemble a pause. However, if the locations of the P waves are followed, they should occur at expected times approximately. Normally, PACs are a fairly common finding on ECGs and usually do not require further investigation.

- Premature Ventricular Contractions (PVC): Premature Ventricular Contractions are early heartbeats that originate in the Purkinje Fibre region of the ventricles. They are extra, abnormal beats. On an ECG, PVCs are seen as wide and bizarre QRS complexes that are not preceded by a P wave. The QRS complex in PVCs is longer than 120 milliseconds and there is a compensatory pause before the next beat. PVCs are rarely dangerous on their own, unless they are frequent, i.e. if they occur more than 10 to 30 times per hour, or if they occur every beat in a row, they can be diagnosed as ventricular tachycardia, which is critical for stroke.

1.2 Goal

The main goal of this thesis is to investigate and compare different transformer architectures with non deep learning (based on extracted features) and other deep learning approaches, e.g. Convolutional Networks, for the specific classification of Sinus Rhythm, Atrial Fibrillation, Atrial Flutter, Premature Atrial Contractions and Premature Ventricular Contractions. The main advantage of the Transformer models is the attention mechanism which is designed to learn relationships within data. This could be useful for tasks such as classifying Atrial Fibrillation in ECGs, where relationships within the data could lead to a better understanding of underlying patterns that depend on specific events. A Convolutional Network focuses on local patterns and is less suited to understanding relationships within extracted features. This work explores this mechanism to extract the harder to detect abnormalities, such as distinguishing between Atrial Fibrillation and Atrial Flutter, or capturing Premature Atrial Contractions and Premature Ventricular Contractions, which relate to different events within the repolarisation and depolarisation cycle within the ECG. Premature Atrial and Ventricular Contractions often occur only once on an ECG. Compared to convolutional filters, Transformer models use the weight-based attention mechanism that can reinforce the model to attend to specific parts of the input. The research question is whether the Transformer model with its attention mechanism can capture these patterns better than a traditional Convolutional Network. In addition, different Transformer

architectures are investigated and analysed, i.e. input preparation, applied positional encoding, number of blocks and heads, attention dimension and further hyperparameter tuning. For the evaluation of the models the Physionet 2021 challenge data [28] and the provided MyDiagnostick databases will be used. The work will first evaluate the models on the Physionet 2021 challenge data itself and then attempt to transfer the pre-trained models from the Physionet 2021 challenge data to the MyDiagnostick database. Necessary pre-processing steps and considerations are discussed. Although the Physionet 2021 challenge data provides 12-lead ECGs, the thesis will limit the experiments to single-lead ECGs to narrow the problem statement and because the provided MyDiagnostik dataset contains only single-lead ECGs. Four research questions are formulated, which will be addressed in the experiments and answered by the end of this thesis:

1.2.1 Research questions

1. How well does a Transformer-based model perform on the Physionet 2021 challenge data compared to a feature-based model or a Convolutional Network?
2. Which model performs best at discriminating SR, AF, AFL, PAC and PVC on both datasets?
3. Can an ensemble Transformer model and Convolutional Network effectively capture spatio-temporal information and improve accuracy?
4. What are the challenges in transferring the pre-trained models from the Physionet 2021 challenge data to the MyDiagnostick database? Do the models generalise well, even though different ECG devices were used?

1.3 Chapter overview

Chapter 1 provided an introduction to the problem statement and research objective of this thesis. Chapter 2 discusses related work in this area. Chapter 3 explains the relevant mathematical background and presents the own approaches. Chapter 4 discusses the experiments and evaluates the models. Chapter 5 summarises the results and gives an outlook for further research.

Chapter 2

Related work

This chapter discusses related work. It is divided into three sections. The first section discusses related work on Transformer-based models for the general classification of various abnormalities in ECGs. The second section discusses related work on models developed for the Physionet 2021 challenge, which provides an annotated dataset for the classification of 26 different arrhythmia types. The final section discusses related work in the specific area of classifying Sinus Rhythm, Atrial Fibrillation, Atrial Flutter, Premature Atrial Contraction and detection of Premature Ventricular Contraction based on deep-learning models.

2.1 Related work on the developed models for the CinC Physionet 2021 challenge

The winning paper of the Physionet 2021 challenge [23] by Nejedly et al. proposes a deep residual CNN network with an additional multihead attention layer. The CNN layer uses large convolutional filters, i.e. 15x15 on the first CNN layer and 9x9 on subsequent CNN layers. Furthermore, the model is designed for 12-lead ECG classification, while for fewer lead configurations the unused leads are padded with zeros. In addition, the authors propose a training loss function specifically designed to meet the challenge evaluation test metrics. Furthermore, data augmentation is applied, although the authors do not describe the methodology behind this in detail. However, the authors have published a follow-up study [24] in which they investigate the model architecture with and without a multihead attention layer. Based on the experiments, they find that the multihead attention layer does not significantly improve the performance. Without the multihead attention layer, their model achieves an overall accuracy of 58% and with the multihead attention layer 57% on the Physionet 2021 challenge data. The second winning paper by Han et al. [15] achieves between 55 and 58 % accuracy on the 2, 3, 4, 6 and 12-lead tasks. In their approach, they use a deep convolutional network to which they concatenate demographic features (age and gender) in the dense layer of the output and use a constant weighted binary cross entropy as loss function. The authors address the goal of achieving high generalisation performance by using a cross validation strategy called "leave-one-dataset-out-cross-validation", which treats each of the seven challenge datasets as one fold, so that one dataset is in the test set, one in the validation set, and the rest are used for training. In addition, the authors use a special data augmentation method called "Mixup" [37]. Mixup makes the model's decision boundary smoother through a regularisation technique that mixes two input samples with their features and labels based on a coefficient. The third winning paper of the Physionet 2021 challenge by

Wickramasinghe and Athif [35] proposes two Convolutional Networks with four residual blocks working in parallel and which achieves an accuracy of 51%-55% on the final test set. One model receives the ECG signal itself as input. The authors apply standard pre-processing steps such as normalisation, resampling and zero padding. For the second model, the authors apply a Fast Fourier Transformation (FFT) to the ECGs to obtain the frequency domain. One model receives the time domain (the pre-processed ECG) as input and the other the frequency domain. Both models are combined by a pooling layer, which reduces the feature space that is then fed into a common dense layer for the final classification output on the 26 classes using a softmax layer.

2.2 Related work for the classification of various arrhythmia types based on Transformer models

Choi et. al [6] propose an encoder-based ECG model, called ECGBERT, which is based on the model architecture and training design of BERT (Bidirectional Encoder Representation from Transformers). Their framework design allows the authors, similar to BERT, to fine-tune their model on multiple down-stream tasks, including atrial fibrillation classification (binary), heartbeat classification (normal, unknown, supraventricular ectopic, ventricular ectopic and fusion heartbeat groups), user verification by using the ECGs as biometric authentication to predict patient IDs and sleep apnea detection. In summary their approach uses two types of inputs to learn high-level relations based on the Transformer-encoder architecture within the ECG. The first input to the Transformer module include CNN embeddings that contain features obtained from an U-Net [31]. The CNN embeddins are fed in the same order as present in the raw ECG. For the second input the authors propose a tokenizer-like technique that creates a sentence of beats, which maps the ECG signal into a pre-defined set of tokens (vocabulary index) that represent specific clustered wave segments. The vocabulary index is categorized into 70 different wave clusters, composed of 12 P wave clusters, 19 QRS complex clusters, 14 T wave clusters and 25 background wave clusters, whereby the background waves belong to segments in between the detected P, QRS, T onsets and offsets, e.g. PR segments or ST segments. The authors apply time-frequency analysis and the Hamilton algorithm [14] to clean the ECG signals and extract onset and offset points from the ECG. Depending on the onset and offset points the authors use four (P, QRS, T or background wave) K-means classifier that employs Dynamic Time Warping (DTW) to cluster and assign the the extracted wave segments into the proper corresponding cluster. Based on the cluster each wave segment is mapped and encoded to a token. From this the authors create ECG sentences and training examples, where ECG sentences are formulated based on the tokens with up to eight consecutive heartbeats. Each heartbeat consists of several wave segments encoded as tokens. The authors do not give detailed information about how the tokens are encoded in detail, i.e. whether these are mapped to random initialized embeddings. Both inputs, the ECG sentence consisting of token embeddings and the CNN feature embeddings are then fed into the Transformer module. The authors add a [sep] token that marks the end of the token sequence. Positional encoding is added to both types of embeddings to retain information about the order. The Transformer is pre-trained similar to BERT using Masked Language Modeling (MLM) by randomly masking 15% of the input tokens. The authors use then the pre-trained model to fine-tune it on several down-stream tasks by adapting output layer. Zhao [39] lists in a review of 2023 Transformer-based models for ECG classification. Most of the Transformer-based ECG models discussed in the paper focus on short-term ECGs by combining the encoder block of a Transformer with a Convolutional Network [3]. [5] [9] [20] [22] [39]. This is because CNNs have a limited receptive field, which prevents it from learning distant dependencies and can rather extract local patterns. On the other hand, Transformers

can learn long-range dependencies through their attention mechanism. The combination of both can capture spatio-temporal information from the ECG signal [39]. Hu et al. [20] propose a more complex architecture that first extracts features from a single-lead ECG signal using multiple CNN layers. Instead of merging the feature maps through a global pooling layer, the authors treat each feature map as an input token to a Transformer encoder and decoder block, similar to the original Transformer architecture [1]. In addition, the authors add positional encoding. The decoder block unmasks each token by token (here the positionally encoded feature maps), through which the decoder block sequentially classifies the next ECG segment. The paper uses the MIT-BIH arrhythmia database [13], which contains annotations for each heartbeat (Normal, Ventricular, Supraventricular, etc.). Bing et al. [3] propose an encoder only Transformer architecture that combines a Vision Transformer (ViT) with a Convolutional Neural Network, called ConVit. Vision Transformer (ViT) [10] was one of the first papers that effectively applied Transformer models to computer vision tasks. In Vision Transformers an embedding is represented as a subpart/pixel group from an image and the model extract global relations within the whole image for classification by using the attention mechanism to focus on important relations.

2.3 Related work for the specific classification of SR, AF, AFL, PAC and PVC based on deep learning models

Wang et al. [34] analyse in their paper an approach, called PVCNet, for the detection of PVC. The authors present a deep CNN network combined with several dense layers. The input of the model is the full sequence of a single-lead ECG. The authors use the publicly available MIT-BIH database [13] for pre-training and test it on the St. Petersburg INCART database. Their model is able to achieve an accuracy of 98% on the test set. In their work, the authors highlight the importance of several pre-processing steps, including data augmentation to overcome class imbalance, splitting the training and validation sets by patients, ECG resampling and filtering, i.e. a butterworth band-pass filter to retain the main frequency components of the signal. Li et al. highlight in their paper [17] the importance of improving the accuracy of assistive ECG devices. They find that the device makes about 18%-24% false positives after revisiting the labels. In their work they develop a deep learning approach that classifies AF, PVC and PAC from the first two leads of an ECG. Their approach uses a combination of CNN and LSTM. In the first stage, the CNN model extracts features, namely the P, QRS and T intervals. The extracted features are then fed into an LSTM model, which classifies the successively extracted features. The interaction between the two models ensures that spatial-temporal features are extracted. With their approach the authors are able to increase the accuracy compared to the device from 0.77 to 0.86 (AF), 0.76 to 0.84 (PVC) and 0.82 to 0.87 (PAC). Another study by Marco et al. [7] focuses on the classification of non PVC and PVC using only the QRS complex. The authors use the MIT-BIH database [13] by extracting all QRS complexes from long-term ECGs in the MIT-BIH database. In their study they compare several models, e.g. random forest, LSTM, bidirectional LSTM, ResNet-18, MobileNetv2 and ShuffleNet. ResNet-18 is a deep convolutional network with 18 layers. MobileNetv2 [32] and ShuffleNet [38] are efficient CNN networks designed to work on mobile devices. Ribero et al. show in their work [29] and [30] that the classification of Atrial Fibrillation and Atrial Flutter is not a trivial task. In their paper, the authors propose a 1D and 2D Convolutional Network trained on 1D ECG signal and 2D ECG image data. The underlying datasets are a combination of the Physionet 2021 challenge data and private datasets collected from various cardiologists and hospitals. However, in their paper [30] the authors state that the performance of the model trained on only the Physionet 2021 challenge data is significantly lower than that of the model trained on their own dataset due to quality. Wang [33] investigates

the classification of Atrial Fibrilltion and Atrial Flutter and proposes a combination of a CNN with an improved version of the Elman Neural Network (IENN) [12], a specific form of the RNN architecture that uses a context node for improved contextual memory. The combined model is able to achieve around 99% accuracy on the MIT-BIH [13] database for the binary classification problem.

Chapter 3

Methods

This chapter discusses the approaches developed for this thesis. It is divided into three sections. The first part discusses the mathematical background of Transformer models by comparing the architectural differences between a Transformer model trained on language and one trained on ECG signals. The second section gives an overview of the main components of a Convolutional Network. The third section presents the implemented approaches that are used to address the problem of ECG classification for 26 different arrhythmia classes, here a feature-based classifier, a Transformer model, a Convolutional Network and several ensemble models.

3.1 Transformer (Mathematical background)

The Transformer model was introduced in the paper "Attention Is All You Need" by Vaswani et al. [1]. Figure 3.1 shows the model architecture.

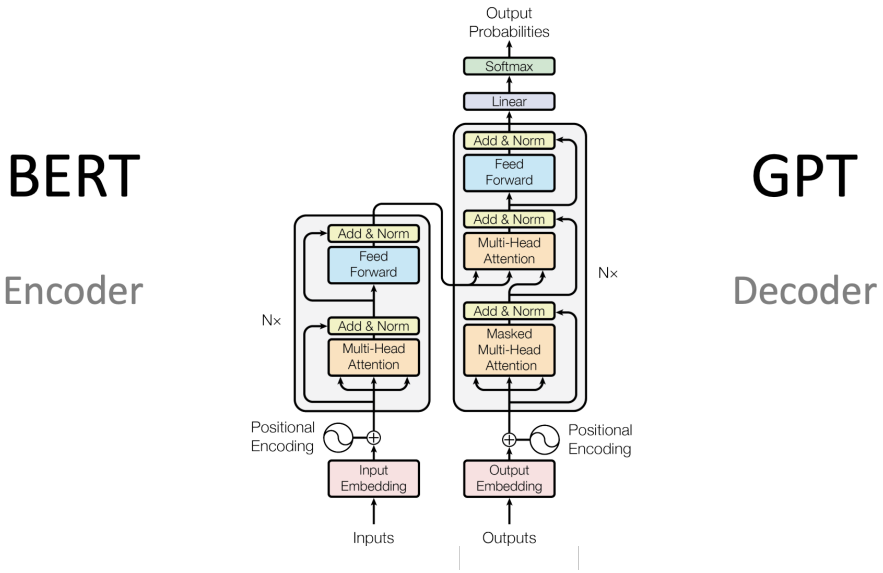

Figure 3.1: Transformer architecture
Source: "Attention Is All You Need" [1]

The model consists of two parts, the encoder block, left half in 3.1, and the decoder block, right half in 3.1, connected by the arrow in the middle. Initially, this architecture was designed for language translation. The encoder block receives the input sequence and encodes the words (tokens). The encoder block maps then the input sequence to the translated output sequence via the decoder block, which predicts the next most likely output token. The main strength of the Transformer model is the "attention mechanism" implemented in the "multi-head attention" block 3.1. The attention mechanism might be compared to a convolutional filter, although it has a different objective and uses a different approach. Similarly, the attention block applies a "filter" that merges each feature with the surrounding features in the input. However, the objective of the Transformer model is to learn relations within the input based on weighted attention, unlike the Convolutional Network, which aims to extract invariant features from local input and feature groups. Therefore, depending on the kernel size, a convolutional filter applies only to local input and feature spaces, while the attention block applies to the entire input sequence. The attention block computes query, key and value pairs by considering each input element individually, depending on the current query (self-attention) and all other input subspaces. During training, the attention block adapts its weights to different arrangements in the training data, i.e. it sees different variations and can extract dependencies and relationships from various training example arrangements. The attention mechanism is discussed in more detail in the next section. Many researchers have adopted the original Transformer architecture and use blocks of it for various downstream tasks, such as (text) classification tasks, e.g. BERT (Bidirectional Encoder Representation from Transformers) [8], or for (text completion) regression tasks, such as GPT (Generative Pre-trained Transformer) [4]. BERT uses only the encoder block, while GPT uses only the decoder block. This thesis implements only the encoder block of the Transformer model. The reason for this choice is the following is that the attention block in the decoder adds masking to the embeddings. More specifically, the encoder focuses its computations on all input embeddings (bidirectional), whereas the decoder is designed for prediction tasks and focuses its computations on the output of the encoder block and on its own past states (unidirectional) by masking future states. This makes the decoder less suitable for classification tasks and could prevent a classification model from learning a more meaningful representation from the entire ECG because valuable information would be masked. Therefore, this works investigates an encoder-only Transformer-based model. In general, a Transformer model divides the input sequence into subspaces, called embeddings. Embeddings are vectors that represent specific input features. In language-based Transformers, each embedding represents an assigned sub-word or character in the known vocabulary of the model, also called tokens. The input preparation process for language-based Transformers work differently as in this work and is handled by a tokeniser, a separate sequence mapping and vocabulary indexing tool of the model. Tokenizers are one of the key differences between Transformers trained on language tasks and Transformers trained on ECG signals. More about tokenizers is shortly discussed in the section 3.1.4. In this work, the Transformer model does not utilize a tokeniser to map the ECG signal. One reason is that the signals are already present as processable numbers and do not need to be encoded. The other reason is that words in language are repetitive, whereas heartbeats or segments of ECG signals have all different shapes and are unique, making it superfluous to index these or train an input embedding matrix that can only represent a finite feature space of arrhythmia embeddings. However, it might be an idea to cluster related segments of arrhythmia beats and map these to a finite set to reduce the variance for training an Transformer model. However, this will not be part of the analysis in this work. In this work, simpler signal pre-processing steps are applied before the ECG data is passed to the model. This includes splitting the signal into segments, e.g. based on RR intervals or signal independent fixed-size segments. The entire single-lead ECG signal can also be fed into the attention layer without any splitting. This will be analysed

and discussed in the evaluation 4.2 and discussion 4.3. The input size in a Transformer model is fixed-size, also known as the context window of the Transformer model. In this work, this is a 10-second single-lead ECG signal. The ECG embeddings are fed directly into the encoder block, whereas in language-based Transformer models an additional trainable embedding matrix represents the first layer of the model before the encoder block.

3.1.1 Positional encoding

The original Transformer model adds positional information to each embedding before these are passed to the attention block. After the positional encoded information is added, the layer is connected with the output of the attention block via a residual connection. This ensures that temporal information about the order of the embedding sequence is preserved and propagated throughout the network. The original paper proposes a positional encoding technique using sine and cosine functions of different frequencies:

$$\mathrm{PE}(pos, 2i) = \sin\left(\frac{pos}{10000^{\frac{2i}{d_{\mathrm{model}}}}}\right) \tag{3.1}$$

$$\mathrm{PE}(pos, 2i+1) = \cos\left(\frac{pos}{10000^{\frac{2i}{d_{\mathrm{model}}}}}\right) \tag{3.2}$$

Pos represents the index position of each embedding in the sequence and i is the embedding dimension, which is of equal size for all input embeddings. Each dimension of the positional encoded embedding corresponds to a single sinusoid and the wavelengths form a geometric progression from 2π to $10000 \cdot 2\pi$. The authors chose this function because they hypothesised that it would allow the model to learn and attend to relative positions more easily, since for any position k, $\mathrm{PE}_{\mathrm{pos+k}}$ can be represented as a linear function of $\mathrm{PE}_{\mathrm{pos}}$. [1]. This thesis investigates the effect of adding positional information to ECG classification and discusses this in more detail in the evaluation 4 and discussion 4.3 sections.

3.1.2 Attention mechanism

The input of the attention block is the sequence consisting of several positional encoded embeddings (features), here a matrix of the size $sequence\_length \cdot embedding\_dimension$. The attention block uses the query, key and value matrices for further processing, denoted by the matrices $\mathrm{W^Q}$, $\mathrm{W^K}$ and $\mathrm{W^V}$. The multiplication of the input $sequence\_length \cdot embedding\_dimension$ with $\mathrm{W^Q}$, $\mathrm{W^K}$ and $\mathrm{W^V}$ projects the input sequence into three different transformed spaces separately, yielding the matrices Q, K and V that go into the attention block, which are represented by the three arrows 3.1. An important advantage here is that the weights $\mathrm{W^Q}$, $\mathrm{W^K}$ and $\mathrm{W^V}$ can be computed in parallel during the forward/backward pass and do not depend on past states as in LSTMs or RNNs. The next steps form the core of the Transformer model, which is the scaled dot-product attention, shown in figure 3.2 and formula 3.3. The first step in this process is to compute the dot-product between Q and K by multiplying Q by $\mathrm{K^T}$, which gives an unscaled attention weight matrix. The shape of the attention weight matrix is $(sequence\_length \cdot sequence\_length)$ (the quadratic number of inputted embeddings). This matrix represents how much each embedding should relate to each embedding and which is used to project this information into the value matrix. The attention scores are divided by $\sqrt{d_k}$, which helps to reduce the variance of the computed relational values from Q and K and stables training. Next, each row in the attention weight matrix is scaled by a softmax function to obtain normalised and interpretable weights. The scaled weights are the final attention weights that will be used to scale the value matrix

and are multiplied with the value matrix V. After the calculation of V follows a linear matrix 3.3. This linear matrix (which should not be interchanged with the feed-forward layer) is also a learnable weight matrix that weights the concatenated heads, which will be discussed in the next section 3.1.3. Usually, the dimensions of the output value matrix V has the same dimension as the input matrix $sequence\_length \cdot embedding\_dimension$. Depending on the dimension configuration of W^Q, W^K and W^V, the initial shape might be restored by this linear matrix operation. The masking operation (optional) 3.2 is skipped here because it is only used in the decoder block. The masking operation adds triangular masking attention matrix e so that the processed attention scores depend only on unmasked/past embeddings. The output of the encoder block is added with the input embeddings by a residual connection. Layer normalisation is applied here to stabilise the weights for training. Next, a feed-forward layer follows the attention block 3.1. The feed-forward layer allows the model to learn non-linearity, since the operations inside the attention block remain linear matrix multiplications. In the original paper a single hidden layer with ReLU activation function is used [1]. For comparison, GPT-3 uses a feed-forward layer with a single hidden layer of 24 neurons. The output of the feed-forward layer is then reconnected by a second residual connection and normalised. The output of the entire encoder block has the same shape as the input ($sequence\_length \cdot embedding\_dimension$), allowing to stack multiple encoder blocks without modification, where each block can learn different relations among the features.

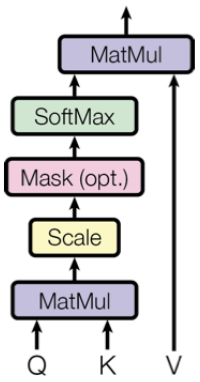

Figure 3.2: Scaled Dot-Product Attention
Source: "Attention Is All You Need" [1]

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{3.3}$$

3.1.3 Multi-Head attention

In addition, the original Transformer introduces a mechanism called Multi-Head Attention, referred to as heads in the paper [1]. The idea of Multi-Head Attention is to train multiple attention matrices on different input sub-spaces from the embeddings simultaneously. This allows to capture a wider range of relationships and to encapsulate partial information from different contexts within the same training example. Multi-head attention splits each embedding into n sub-embeddings before feeding these to h query, key and value matrices in parallel. In the original paper 8 heads were used [1]. For example, a Transformer trained on 512-dimensional input embeddings, each embedding is splitted into 8 sub-embeddings, yielding $sequence\_length \cdot (8 \cdot 64)$

dimensional sub-embeddings as input to the attention block, where each of the 8 sub-embeddings is separately passed to an attention head. The computation then proceeds as described in the previous section about the scaled dot-product attention, but here in parallel on different embedding sub-spaces and separate W^Q, W^K and W^V matrices. After the computation of 8 different value matrices, these are concatenated and multiplied by a learnable linear matrix. The linear layer ensures that the initial size of $sequence\_length \cdot 512\_dimensional\_embeddings$ is restored, even when dimension_query_key_or_value * heads does not match the embedding_size. On top it serves as a further trainable linear projection (without the use of an activation function) to not simply concatenate the information obtained from different heads. Figure 3.3 illustrates this process.

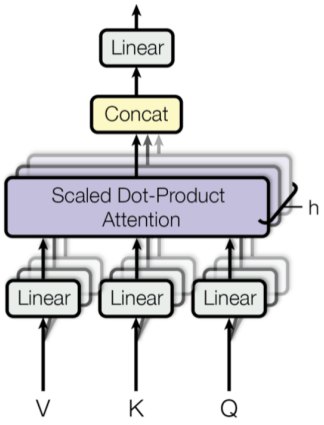

Figure 3.3: Multi-head attention
Source: "Attention Is All You Need" [1]

3.1.4 More about Transformers, Tokenizers, BERT and GPT

This section is intended to give some task-related capability information and ideas on how the model might be adapted, but which will not be further investigated in this work.

In recent years, Transformer models have gained considerable popularity because their model and training design can be much more easily accelerated on a graphic processing unit (GPU) / graphic card, due to the property of shared weights during inference and capability parallel weight updates during backpropagation through the attention (query, key and value) matrices. This leads to a significant reduction in training time and hardware costs, while maintaining comparable or even better performance, compared to the previous Long-Short-Term-Memory (LSTM) [16] or Recurrent Neural Network (RNN) [19], which were widely used until Transformer models for sequential data. LSTMs and RNNs must process data sequentially through their units and do not scale well for large training applications due to slowness and computational costs. In addition, Transformer models have the ability to process long-range sequential data much better, because LSTMs and RNNs can suffer from long-range dependencies caused by vanishing gradients during backpropagation.

In Transformers trained on language, the tokeniser is an important step that splits the input sequence into sub-words and characters that are mapped to a predefined entry in the input em-

bedding matrix of the model. The embedding matrix is a trainable vocabulary matrix index as part of the model, whereas the tokeniser is not part of the model and serves as a seperate mapping tool. The reason that Transformer models split the input sequence into sub-words and characters is that it can drastically reduce the amount of training required. Much more training would be required with an embedding matrix contains every written word. For example, A special token in a tokeniser handles all adverbs. Instead of representing each adverb in the model, the tokeniser treats "-ly" separately as an individual token, thus every verb only needs to be indexed in its base form. This resulted that the tokeniser in BERT has only a vocabulary size of around ∼30000 [8] tokens. Other special tokens include seperation, padding and out-of-vocabulary tokens (representing unknown words).

Researchers have proposed various strategies to modify the transformer architecture, e.g. BERT [8]. BERT is specifically fine-tuned for text classification and Q&A tasks. BERT is designed so that it can be used for classification and regression tasks at the same time. BERT uses only the encoder block to learn a contextualised representation from the input sequence. In the original paper, BERT stacks 12 encoder blocks [8], where each block is intended to learn other relations and adds incrementally more contextual information to the outputted embedding. For text classification tasks, BERT also outputs a separate [CLS] token, which is obtained through a dense layer from the attention block and can be used to train the model on classification tasks. For Q&A tasks, the model is adapted to receive two sequences as input by adding segment information to the embeddings (i.e. either segment one or two) after the positional encoded information has been added. Furthermore, an unique separation token [SEP] is entered to mark the separation of the two sequences, allowing the model to be trained on a pair of sequences. The training process of BERT has been done by two tasks, Masked Language Modelling (MLM) and Next Sentence Prediction (NSP) [8]. Both MLM and NSP mask random embeddings or a segment in the input, similar to text generation, but by predicting a whole segment and not a single token. Researchers continued to adapt BERT for other tasks. For example, thhese include semantic search and semantic textual similarity, proposed by Reimers et al. in Sentence-BERT (S-BERT) [27]. S-BERT either processes a single sequence and uses a feed-forward and pooling layer on top that denses the output to a single embedding, representing the entire input sequence. The encoded representation of the sequence can be used for fast semantic search, where the outputted embedding can be compared by dot-product or cosine distances with other encoded sequence embeddings, commonly used in vector databases. Semantic textual similarity takes two sequences and performs "cross-attention" on the sequence pairs. It is fine-tuned to directly output a similarity score obtained by a dense layer, which is trained on sequence pairs and correlation scores.

GPT [4] adopts only the decoder block. The scaled dot-product attention computation adds triangular matrix masking to the sequence. During training, GPT sequentially unmasks and predicts future embeddings for generation tasks using a sliding window. The model is then aligned to specific prompts, allowing it to learn more abstract patterns within text generation specific tasks, such as chat conversations, question answering, text summarisation, enumeration or multiple choice tasks. Moreover, GPTs training is combined with a reinforcement learning strategy that uses word sampling and penalty techniques during inference to allow the model to produce varying outputs, which are then manually ranked by annotators. In general, GPTs performance has been achieved by training the model on large text data. For example, GPT-3 uses 96 decoder blocks with an embedding size of 12,288, a context window of 2048 and a total of 175,181,291,520 trainable parameters [4]. GPT-4 uses about 1,760,000,000,000 parameters.

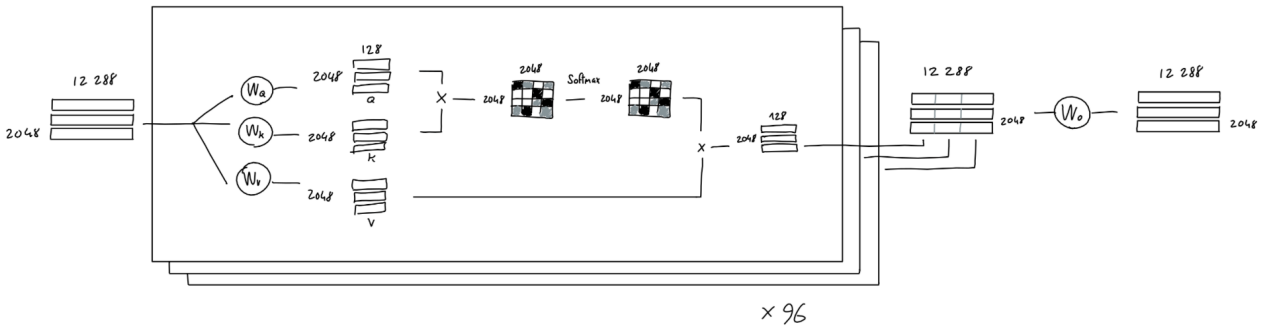

Figure 3.4: Attention mechanism & Multi-Head attention together

3.2 Convolutional network (Mathematical background)

Convolutional networks are successful and frequently applied deep learning based models to the classification of ECG signals and [17] [7] [34], including the models of the three winning papers from the Physionet 2021 challenge [23] [15] [35]. Initially designed for image classification, these models can also be applied to 1D ECG signals. The main component in a convolutional network is a convolutional filter, a fixed-sized window (e.g. 3x3, called kernel) that stripes over the input data, where each input data point and its local neighbor data points are multiplied with the kernel. The output is a new data point, which is a weighted average from the current data points multiplied with the filter. These filters are the trainable weights in an convolutional network that learn specific features from the input. At the beginning the weights are randomly initialized. The transformed data is called feature map, which represents an obtained feature space from the input. Several convolutional filters can be applied in parallel in the same layer, yielding several features maps as output. A convolutional layer is typically configured by the number of convolutional filters, kernel, stride and padding size. The number of filter in CNNs determines how many distinctive convolutional filters (and resulting feature maps) are trained in the same layer. Usually, this number starts small (e.g. 32) and doubles from one layer to the next deeper layer (e.g. 64, 128, ...). The reason for this is that in deeper layers more complex and abstract features are combined from the simpler features of previous layers. This process often requires a larger number of filters to capture the increasing complexity and diversity of features. Feature maps encode different features from the input and increase the dimensionality. However, these featrue maps usually are merged in the final layers of an CNN network, which will be discusses in a bit. The filter size determines the number of trainable parameters, smaller can prevent overfitting, enables deeper and more complex networks. Smaller filters can focus on details, while larger filters can recept distant associated patterns. Stride determines the number of data points the kernel is shifted when sliding over the input data. Padding adds margin to the borders of the input and can reduce the input size. An convolutional layer a normalization layer can follow, which normalizes the output values. This prevents exploding gradient problems, since it avoids infinity large becoming weights and enables proper training. Normally, to reduce dimensionality a pooling layers is applied that follows a convolutional layer. There are three common pooling types: "min", "max" and "average". This means a pooling layer takes either the minimum, the maximum or the average value from a window of data points as new output. On a 1D signal a pooling window size of 2 will reduce the dimensionality by half and of 3 to 1/3. The pooling layer can also be applied across feature maps, which merges the feature maps.

Beside, other common parts of convolutional networks include standard feed forward or dense layers using activation functions such as sigmoid, relu, leaky relu, gelu, and others. This layers can add more non-linearity to the network, as CNNs are based on point wise vector or matrix multiplication. Moreover, by ignoring some weights, called dropout, the network can further prevent overfitting and generalize better.

3.3 Own approaches

3.3.1 Feature-based model

As baseline comparison, a feature-based model is trained based on the Biobss Python package. The Biobss library provides a set of toolkits for extracting features from ECG signals. Table 3.3.1 shows 39 extracted features using the Biobss library and a short description for each feature. In a first step, the Biobss library calculates all R peaks and other fiducial points in the ECG, e.g. P, Q, S and T wave onset and offset points. For the point localisation the package has one of the three methods implemented: "pantompkins" [25], "hamilton" [14] or "elgendi" [11]. As the feature-based model is just a baseline comparison, only the "pantompkins" method is investigated in the experiments. These methods are also the default implemented algorithms for fiducial points localisation in the Neurokit2 package. To mention the Neurokit2 package is a more maintained library to work with ECG data, but which is due to similar methods not further discussed here. In the next step, inbuilt functions in the Biobss library calculate 39 morphological features from the extracted ECG locations (peaks and wave onset/offset points). The Biobss library splits the ECG into segments using the detected fiducial points, where for each segment peak amplitudes, intervals and ratios are calculated as features. The library provides two functions: biobss.ecgtools.ecg_features.from_Rpeaks and biobss.ecgtools.ecg_features.from_waves. "from_Rpeaks" takes as reference four consecutive R peaks and calculates the corresponding (current) R peak amplitude, RR intervals (the RR interval before the current R peak and two RR intervals after the current R peak), ratios and the mean of the intervals (3.3.1, row 1-8). "from_waves" takes as reference two consecutive R peaks and calculates the corresponding P, Q, R, S and T amplitudes, their intervals and ratios (3.3.1, row 9-39). Each feature is an average from all segments in the ECG signal. These features are then used to train several feature-based classifiers, e.g. random forest, adaboost and a support vector machine. For this, the work utilises the Sklearn Python framework. The classifier parameter configurations and results are discussed in section 4 and 4.3.

Biobss features	
Features (averaged)	**Description**
a_R	Amplitude of R peak
RR0	Previous RR interval
RR1	Current RR interval
RR2	Subsequent RR interval
RRm	Mean of RR0, RR1, and RR2
RR_0_1	Ratio of RR0 to RR1
RR_2_1	Ratio of RR2 to RR1
RR_m_1	Ratio of RRm to RR1
t_PR	Time between P and R peak locations
t_QR	Time between Q and R peak locations
t_SR	Time between S and R peak locations
t_TR	Time between T and R peak locations
t_PQ	Time between P and Q peak locations
t_PS	Time between P and S peak locations
t_PT	Time between P and T peak locations
t_QS	Time between Q and S peak locations
t_QT	Time between Q and T peak locations
t_ST	Time between S and T peak locations
t_PT_QS	Ratio of t_PT to t_QS
t_QT_QS	Ratio of t_QT to t_QS
a_PQ	Difference of P wave and Q wave amplitudes
a_QR	Difference of Q wave and R wave amplitudes
a_RS	Difference of R wave and S wave amplitudes
a_ST	Difference of S wave and T wave amplitudes
a_PS	Difference of P wave and S wave amplitudes
a_PT	Difference of P wave and T wave amplitudes
a_QS	Difference of Q wave and S wave amplitudes
a_QT	Difference of Q wave and T wave amplitudes
a_ST_QS	Ratio of a_ST to a_QS
a_RS_QR	Ratio of a_RS to a_QR
a_PQ_QS	Ratio of a_PQ to a_QS
a_PQ_QT	Ratio of a_PQ to a_QT
a_PQ_PS	Ratio of a_PQ to a_PS
a_PQ_QR	Ratio of a_PQ to a_QR
a_PQ_RS	Ratio of a_PQ to a_RS
a_RS_QS	Ratio of a_RS to a_QS
a_RS_QT	Ratio of a_RS to a_QT
a_ST_PQ	Ratio of a_ST to a_PQ
a_ST_QT	Ratio of a_ST to a_QT

3.3.2 Encoder-based Transformer model

To investigate which Transformer configuration fits best for the classification of multiple arrhythmia diseases, several architectures are investigated and compared. Below is a table given with different Transformer configurations. Investigated parameters are number of attention blocks, whether positional encoding is added, sequence length/embeddings size and number of heads. The input length is for all models a 10s long ECG signal (2000 data points). The sequence length/embedding size determines therefore in how many segments the ECG is splitted, e.g. a sequence length of 10 with 200-dimensional long embeddings. All models are trained with 10-fold cross-validation.

3.3.3 Residual CNN

3.3.4 Ensembled models

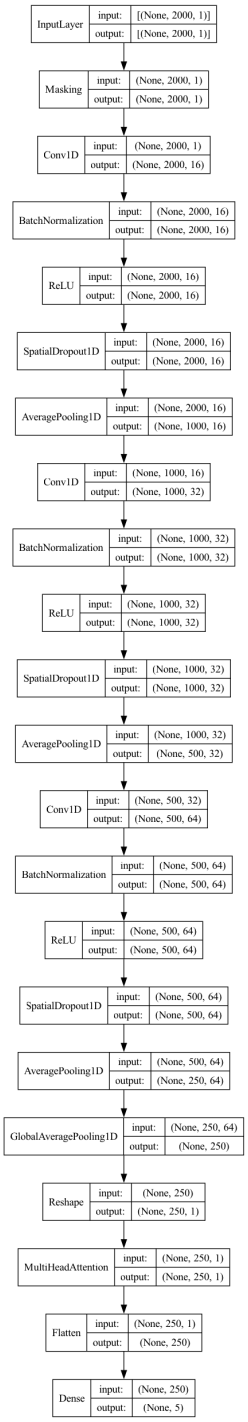

Figure 3.5: Physionet 2021 class distribution - SR, AF, AFL, PAC and PVC

Chapter 4

Evaluation

4.1 Datasets

4.1.1 Physionet 2021 challenge data

This section briefly summarises the publicly available Physionet 2021 challenge data. The Physionet 2021 challenge data provides a diverse real-world database and is a collection of datasets from seven sources in four countries on three continents (Physionet 2021) [28]. The challenge data contains 88,253 12-lead ECG recordings. However, many of the samples have more than one label, resulting in approximately 180,000 samples. The sources of the challenge data include the Ningbo database, PTB-XL database, Chapman-Shaoxing database, Georgia 12-lead ECG challenge data, CPSC and CPSC-extra database, PTB and INCART database (see (Physionet 2021). The table 4.1 shows an overview of the different ECG lengths from the Physionet 2021 training challenge data and their proportion to the entire challenge data [15].

Dataset source	**Average ECG length (seconds)**	**Data samples**
Ningbo database	10s	34,905
PTB-XL database	10s	21,837
Chapman-Shaoxing database	10s	10,247
Georgia 12-lead challenge data	9s	10,344
CPSC database	15s	6. 877
CPSC-extra database	15s	3,453
PTB database	110s	516
INCART database	1800s	74

Table 4.1: Physionet 2021 challenge data composition

This work only uses the published training data, as the test data for the official scoring metrics and evaluation are withheld and not publicly available. Figure 4.1 shows the class distribution for the entire publicly available challenge training data with the arrhythmia class names and Snomed CT codes (a unique ID assigned to each arrhythmia type). The graph shows that the challenge data is quite unbalanced. For example, 28,971 sinus samples are provided, while many classes are present with less than 1,000 examples. To properly train the models, the dataset needs to be pre-processed and balanced to avoid overfitting on some major classes. For the official challenge metrics, a subset of 26 classes was selected (see official scored labels), shown in figure 4.2 (six

classes are combined due to the limited samples in these classes). For the evaluation of the transferred models on the MyDiagnostick data, part of the experiments will focus only on the classification of sinus rhythm (SR), atrial fibrillation (AF), atrial flutter (AFL), premature atrial contractions (PAC) and premature ventricular contractions (PVC). This is due to the limited class annotation in the MyDiagnostick dataset, which consists only of these class labels. Figure 4.3 shows the class distribution of these samples separately. The most underrepresented class is premature ventricular contraction with 1279 samples.

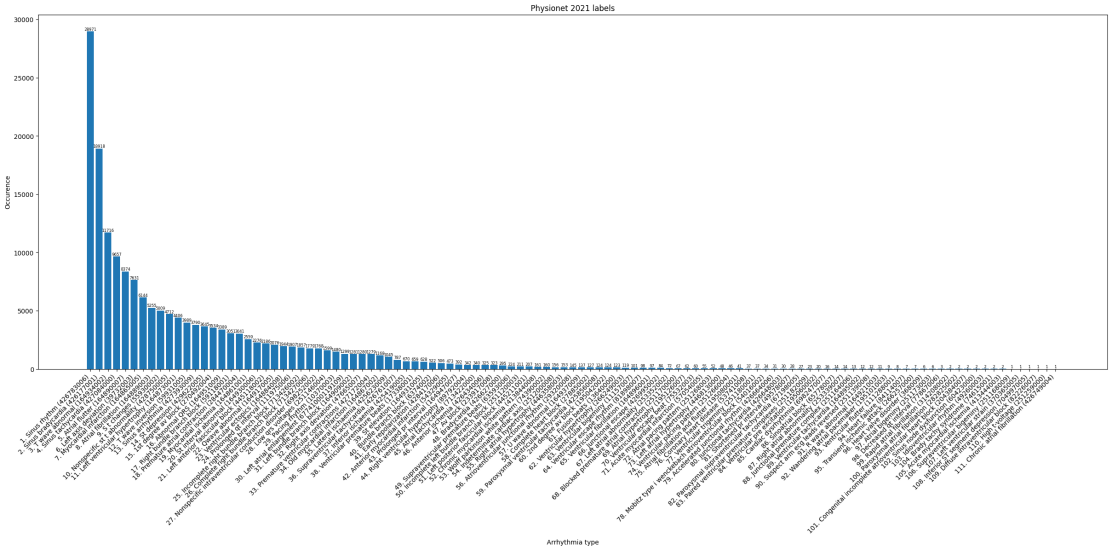

Figure 4.1: Physionet 2021 all classes distribution

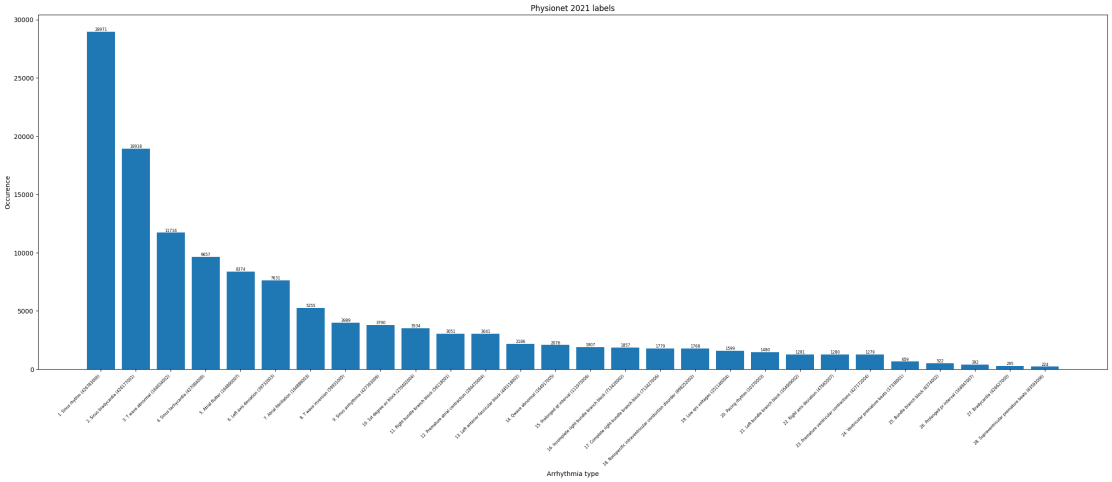

Figure 4.2: Physionet 2021 scored classes distribution

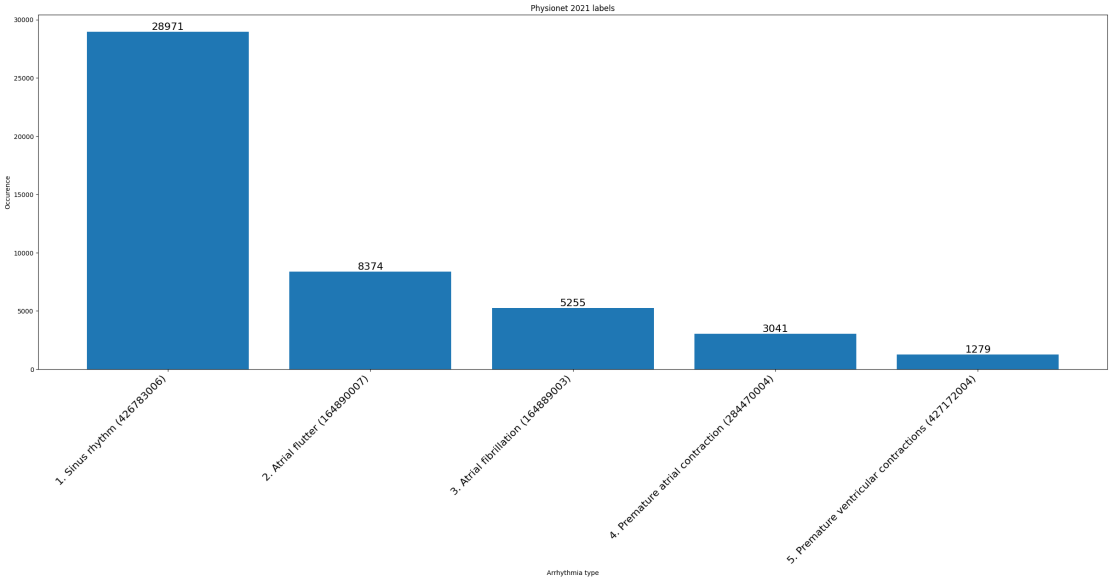

Figure 4.3: Physionet 2021 class distribution - SR, AF, AFL, PAC and PVC

4.1.2 MyDiagnostick data

The MyDiagnostick class distribution is shown in figure 4.4. The dataset consists of more than 40,000 samples, from which only about 11,000 have been manually annotated. The remaining samples are automatically labelled as either sinus rhythm or atrial fibrillation by the assistive ECG device. However, in some of the samples with atrial ventricular contractions and premature ventricular contractions, both classes were present on the same ECG. For simplicity, these samples have been discarded in the evaluation. The aim of the experiments is to evaluate the pre-trained models from the Physionet 2021 challenge data on the manually annotated subset and as part of the thesis to provide annotations for the remaining 29,000 samples based on the most accurately developed and evaluated model.

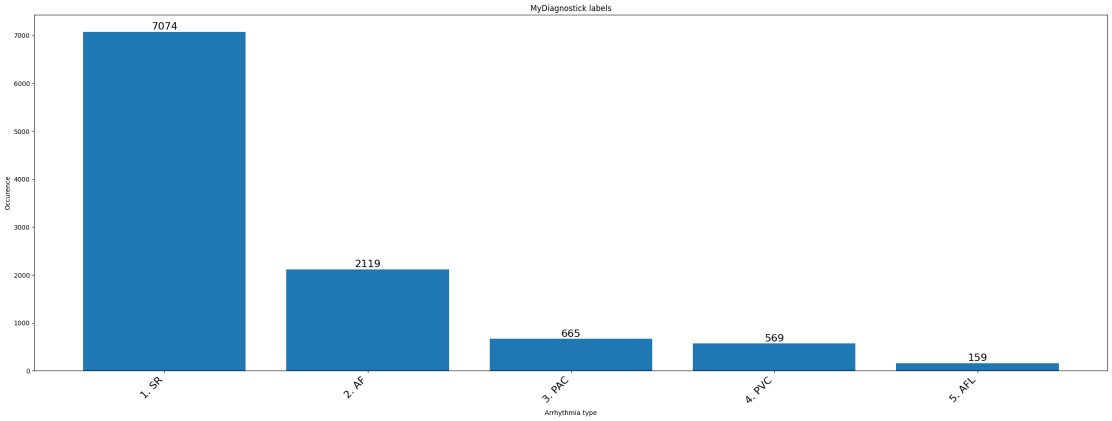

Figure 4.4: MyDiagnostick class distribution - SR, AF, AFL, PAC and PVC

4.1.3 Pre-processing

This section summarises the pre-processing steps applied to both datasets. The preprocessing steps include zero padding/truncation, resampling, normalisation, filtering, split-test by patient, class balancing and data augmentation. Since the Physionet 2021 challenge data consists of 12-lead ECGs and the MyDiagnostick data consists of single-lead ECGs (first lead I), all models in the experiments are pre-trained on the first lead from the Physionet challenge data. The ECGs in the Physionet 2021 challenge data were recorded at a sampling rate of 500Hz, while the MyDiagnostick ECGs were recorded at a sampling rate of 200Hz. Most of the ECGs in the Physionet 2021 challenge data are 10 second recordings (5000 data points for 10 seconds - see 4.1). The MyDiagnostick data contains 60 second ECGs with a length of 12000 data points (2000 data points for 10 seconds). Firstly, each ECG in the Physionet 2021 challenge data has been truncated and zero padded to 5000 data points. Secondly, these ECGs are downsampled to 2000 data points, making both datasets uniform in terms of sample density and temporal resolution. Next, each recording in both datasets was normalised within the range of -1 and 1 using min-max scaling:

$$x' = 2\left(\frac{x - \min(x)}{\max(x) - \min(x)}\right) - 1 \tag{4.1}$$

This is particularly helpful in reducing signal amplitude variation due to differences in electrode placement, body size and individual heart activity. In addition, normalisation helps to improve unbiased and stable training when using gradient descent, as it prevents the model from being biased towards certain input features (e.g. R-amplitude) and avoids gradient explosion problems during backpropagation by keeping the model weights in an uniform range. In the next step, a standard Butterworth bandpass filter was applied to both datasets with a bandwidth of 0.3Hz to 21Hz, which reduces noise and facilitates the learning of key features. In addition, the Physionet 2021 challenge data was split by patient for training (80% training - 20% validation - 20% test) to ensure that the model learns general features rather than specific features of individual patients. As the MyDiagnostick data is used to transfer the scoring from the pre-training to the Physionet 2021 challenge data, it is only used as a test set. As figures 4.2 and 4.3 show, the training data is highly unbalanced. To address this issue, standard class downsampling and upsampling were applied. For the evaluation of the 5 classes (SR, AF, AFL, PAC, PVC) problem models, each class was downsampled or upsampled to 4000 samples, by randomly duplicating some ECGs. The class upsampling was applied after the training and test splits of the patients on the individual subsets, while for the test, to avoid predicting the same ECG several times, only down-sampling was applied. Moreover, this ensures uniform evaluation metrics, i.e. if the model is good in predicting sinus rhythm and there are lot of sinus rhythm cases in the test set, the evaluation would be biased. For the evaluation on the 26 classes from the Physionet 2021 challenge data itself, segment swapping was used. Segment swapping is a data augmentation technique that divides the ECG into segments and randomly swaps some segments.

4.2 Experimental setup and results

This section describes the experimental setup and results of the developed models. The section is divided into two parts. The first part discusses the experimental setup and results for the evaluation of the models on the Physionet 2021 challenge data itself. The second part discusses the experimental setup and results for the pre-trained models on the Physionet 2021 challenge transfered and tested on the MyDiagnostick database. The transformer and convolutional network models have been trained for 250 epochs on a Nvidia T4 (16GB vram) with early stopping enabled.

4.2.1 Physionet 2021 challenge

In this section the results from the evaluated models are discussed.

Input shape	**Positional encoding**	**Encoder blocks**	**Heads**	**qkv dim**	**ff dim**	**Dropout**	**Trainable param.**	**Accuracy**	**Precision**	**Recall**	**F1**
(40, 50)	True	1	1	25	24	0.1	155.375	0.096	0.514	0.120	0.194
(40, 50)	True	1	1	25	24	0.4	155.375	0.065	0.418	0.083	0.139
(40, 50)	True	8	1	25	24	0.1	878.818	0.061	0.579	0.079	0.139
(40, 50)	True	8	1	25	24	0.4	878.818	0.098	0.592	0.122	0.203
(40, 50)	False	1	1	25	24	0.1	155.375	0.227	0.747	0.25	0.374
(40, 50)	False	1	1	25	24	0.4	155.375	0.224	0.742	0.253	0.378
(40, 50)	False	8	1	25	24	0.1	878.818	0.228	**0.765**	0.261	0.389
(40, 50)	False	8	1	25	24	0.4	878.818	0.226	0.737	0.255	0.379
(10, 200)	False	8	1	25	24	0.1	1.004.818	0.160	0.744	0.174	0.283
(10, 200)	False	8	8	25	24	0.1	2.129.018	0.177	0.709	0.197	0.308
(40, 50)	False	8	8	400	24	0.1	6.035.018	0.223	0.762	0.247	0.374
(40, 50)	False	8	8	25	2048	0.1	65.947.210	0.219	0.740	0.257	0.382
(40, 50)	False	8	8	400	2048	0.1	70.819.210	0.226	0.751	0.257	0.383
(40, 50)	False	8	8	400	2048	0.4	70.819.210	**0.245**	0.739	**0.286**	**0.413**

Table 4.2: Physionet 2021 train/test split model comparison

Model	Accuracy	Precision	Recall	F-measure
Random Forest (Biobss features)	0.272	0.800	0.291	0.427
Residual CNN	**0.392**	**0.839**	**0.505**	**0.63**
Standard Transformer Encoder*	0.238	0.733	0.280	0.406
CNN + Attention	0.272	0.768	0.330	0.462
Wavelet + CNN + Attention	0	0	0	0
Spectogram + CNN + Attention	0	0	0	0
Ensemble of Transformer (AdaBoost)	0	0	0	0
Ensemble of various Models	0	0	0	0

Table 4.3: Physionet 2021 train/test split model comparison

Model	AUROC	AUPRC	Accuracy	F-measure	Challenge metric
Random Forest (Biobss features)	0.554	0.146	0.272	0.137	0.073
Residual CNN	**0.895**	**0.477**	**0.392**	**0.359**	**0.376**
Standard Transformer Encoder*	0.740	0.246	0.238	0.163	0.055
CNN + Attention	0.797	0.278	0.272	0.162	0.140
Wavelet + CNN + Attention	0	0	0	0	0
Spectogram + CNN + Attention	0	0	0	0	0
Ensemble of Transformer (AdaBoost)	0	0	0	0	0
Ensemble of various Models	0	0	0	0	0

Table 4.4: Physionet 2021 challenge metric scores model comparison

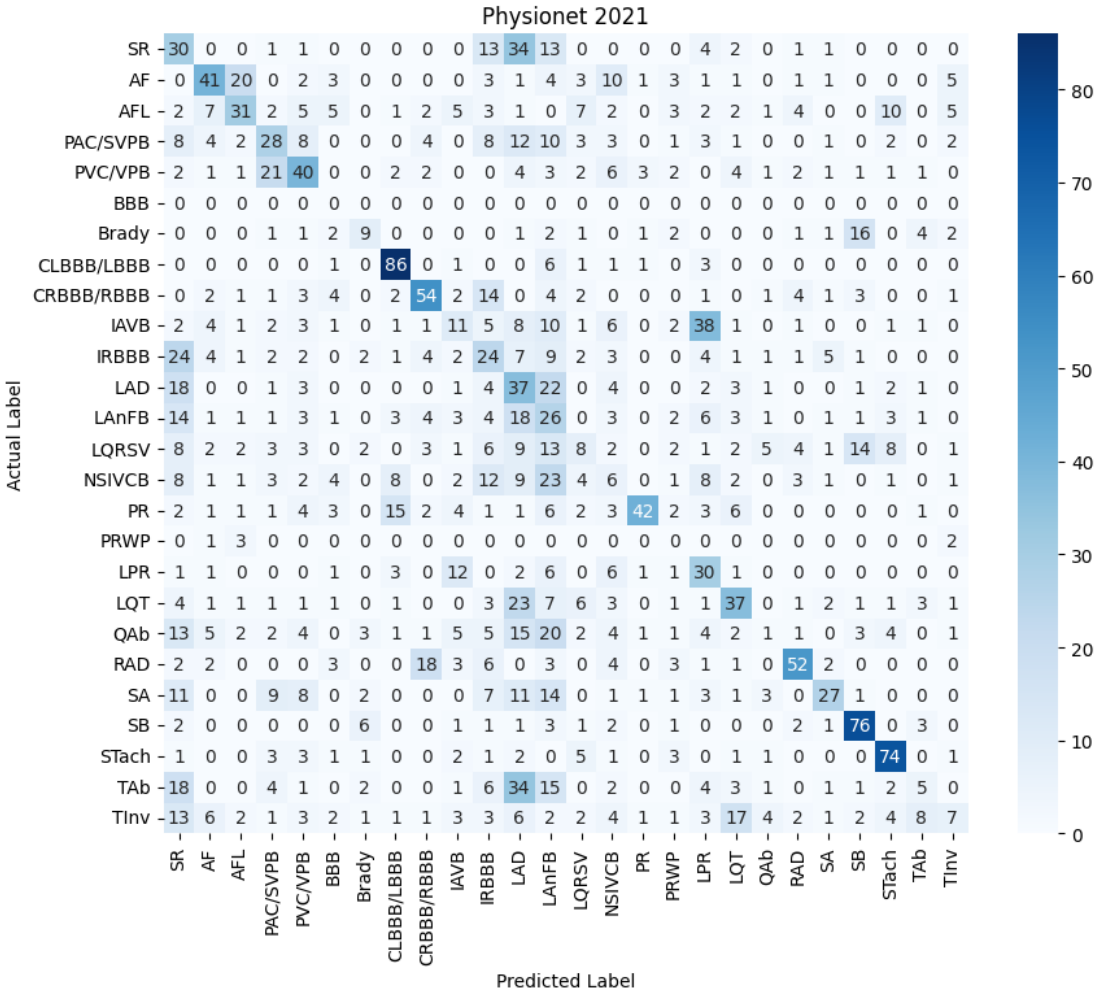

Figure 4.5: Confusion matrix - Residual CNN - scored labels

4.2.2 MyDiagnostick

4.3 Discussion

Model	Accuracy	Precision	Recall	F-measure
Random Forest (Biobss features)	0	0	0	0
Residual CNN	0	0	0	0
Standard Transformer Encoder	0	0	0	0
CNN + Attention	0	0	0	0
Wavelet + CNN + Attention	0	0	0	0
Spectogram + CNN + Attention	0	0	0	0
Ensemble of Transformer (AdaBoost)	0	0	0	0
Ensemble of various Models	0	0	0	0

Table 4.5: Mydiagnostick model comparison

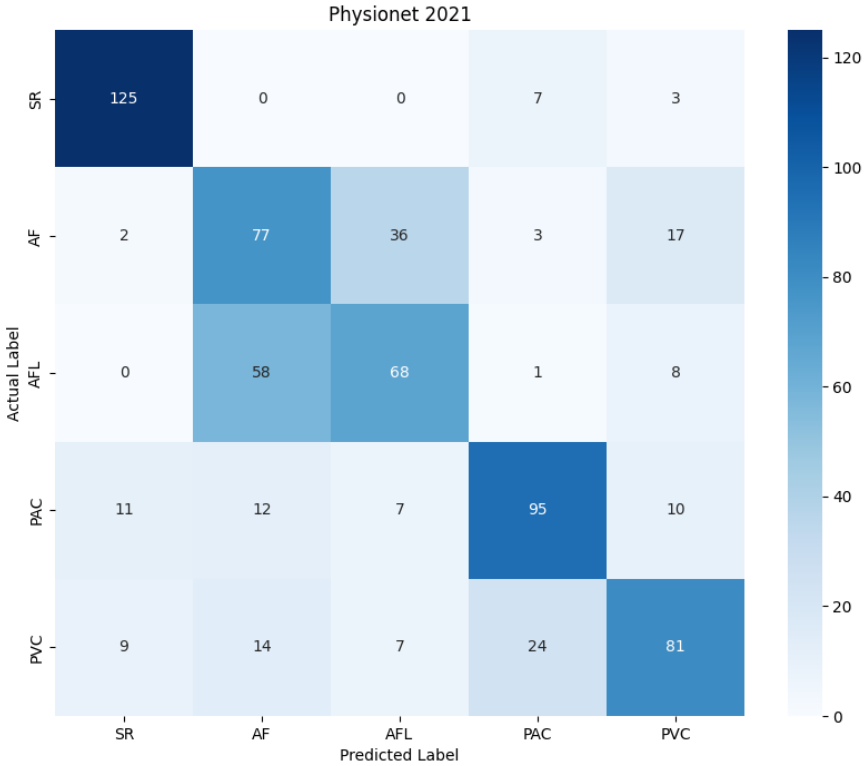

Figure 4.6: Confusion matrix - Residual CNN - SR, AF, AFL, PAC and PVC

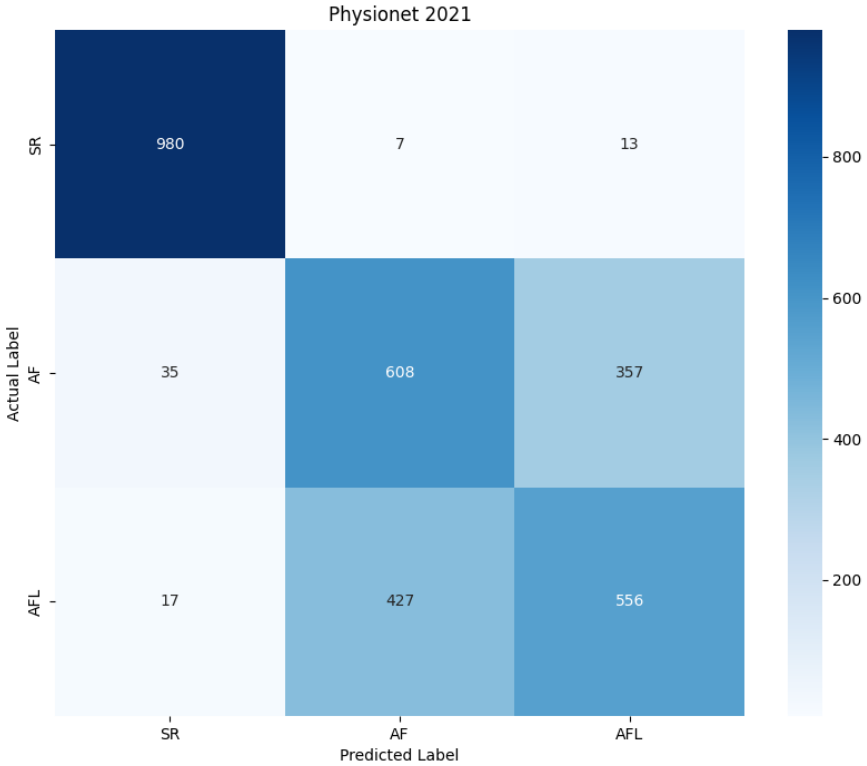

Figure 4.7: Confusion matrix - Residual CNN: SR, AF and AFL

Chapter 5

Conclusion

5.1 Summary

5.2 Outlook

Bibliography

[1] Ashish Vaswani and Noam Shazeer and Niki Parmar and Jakob Uszkoreit and Llion Jones and Aidan N. Gomez and Lukasz Kaiser and Illia Polosukhin. Attention is all you need, 2023.

[2] J. Bender, K. Russell, L. Rosenfeld, and S. Chaudry. *Oxford American Handbook of Cardiology*. 2010.

[3] Pingping Bing, Yang Liu, Wei Liu, Jun Zhou, and Lemei Zhu. Electrocardiogram classification using tsst-based spectrogram and convit. 2022.

[4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[5] Chao Che, Peiliang Zhang, Min Zhu, Yue Qu, and Bo Jin. Constrained transformer network for ecg signal processing and arrhythmia classification. 2021.

[6] Seokmin Choi, Sajad Mousavi, Phillip Si, Haben G. Yhdego, Fatemeh Khadem, and Fatemeh Afghah. Ecgbert: Understanding hidden language of ecgs with self-supervised representation learning, 2023.

[7] Fabiola De Marco, Filomena Ferrucci, Michele Risi, and Genoveffa Tortora. Classification of qrs complexes to detect premature ventricular contraction using machine learning techniques. 2022.

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

[9] Yanfang Dong, Miao Zhang, Lishen Qiu, Lirong Wang, and Yong Yu. An arrhythmia classification model based on vision transformer with deformable attention. 2023.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. 2020.

[11] Mohamed Elgendi, Mirjam Jonkman, and Friso De Boer. Frequency bands effects on qrs detection. 2010.

[12] Jeffrey L. Elman. Finding structure in time. 1990.

[13] Ziti Fariha, Ryojun Ikeura, and Soichiro Hayakawa. Arrhythmia detection using mit-bih dataset: A review. 2018.

[14] P. Hamilton. Open source ecg analysis. 2002.

[15] Hyeongrok Han, Seongjae Park, Seonwoo Min, Hyun-Soo Choi, Eunji Kim, Hyunki Kim, Sangha Park, Jinkook Kim, Junsang Park, Junho An, Kwanglo Lee, Wonsun Jeong, Sangil Chon, Kwonwoo Ha, Myungkyu Han, and Sungroh Yoon. Towards high generalization performance on electrocardiogram classification. 2021.

[16] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. 1997.

[17] Jianyuan Hong, Hua-Jung Li, Chung chi Yang, Chih-Lu Han, and Jui chien Hsieh. A clinical study on atrial fibrillation, premature ventricular contraction, and premature atrial contraction screening based on an ecg deep learning model. 2022.

[18] Shenda Hong, Yuxi Zhou, Junyuan Shang, Cao Xiao, and Jimeng Sun. Opportunities and challenges of deep learning methods for electrocardiogram data: A systematic review, 2020.

[19] J J Hopfield. Neural networks and physical systems with emergent collective computational abilities. 1982.

[20] Rui Hu, Jie Chen, and Li Zhou. A transformer-based deep neural network for arrhythmia detection using continuous ecg signals. 2022.

[21] Melanie Humphreys. *Nursing the Cardiac Patient*. 2013.

[22] Lingxiao Meng, Wenjun Tan, Jiangang Ma, Ruofei Wang, Xiaoxia Yin, and Yanchun Zhang. Enhancing dynamic ecg heartbeat classification with lightweight transformer model. 2022.

[23] Petr Nejedly, Adam Ivora, Radovan Smisek, Ivo Viscor, Zuzana Koscova, Pavel Jurak, and Filip Plesinger. Classification of ecg using ensemble of residual cnns with attention mechanism. 2021.

[24] Petr Nejedly, Adam Ivora, Ivo Viscor, Zuzana Koscova, Radovan Smisek, Pavel Jurak, and Filip Plesinger. Classification of ecg using ensemble of residual cnns with or without attention mechanism. 2022.

[25] Jiapu Pan and Willis J. Tompkins. A real-time qrs detection algorithm. 1985.

[26] Adam G. Polak, Bartłomiej Klich, Stanisław Saganowski, Monika A. Prucnal, and Przemysław Kazienko. Processing photoplethysmograms recorded by smartwatches to improve the quality of derived pulse rate variability. 2022.

[27] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks, 2019.

[28] Matthew A Reyna, Nadi Sadr, Erick A Perez Alday, Annie Gu, Amit J Shah, Chad Robichaux, Ali Bahrami Rad, Andoni Elola, Salman Seyedi, Sardar Ansari, Hamid Ghanbari, Qiao Li, Ashish Sharma, and Gari D Clifford. Will two do? varying dimensions in electrocardiography: The physionet/computing in cardiology challenge 2021. 2021.

[29] Estela Ribeiro, Felipe Dias, Quenaz Soares, Jose Krieger, and Marco Gutierrez. Deep learning approach for detection of atrial fibrillation and atrial flutter based on ecg images. 2023.

[30] Estela Ribeiro, Quenaz Bezerra Soares, Felipe Meneguitti Dias, Jose Eduardo Krieger, and Marco Antonio Gutierrez. Can deep learning models differentiate atrial fibrillation from atrial flutter? 2024.

[31] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[32] Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation. 2018.

[33] Jibin Wang. Automated detection of atrial fibrillation and atrial flutter in ecg signals based on convolutional and improved elman neural network. 2019.

[34] Ziqiang Wang, Kun Wang, Xiaozhong Chen, Yefeng Zheng, and Xian Wu. A deep learning approach for inter-patient classification of premature ventricular contraction from electrocardiogram. 2024.

[35] Nima L Wickramasinghe and Mohamed Athif. Multi-label cardiac abnormality classification from electrocardiogram using deep convolutional neural networks. 2021.

[36] Genshen Yan, Shen Liang, Yanchun Zhang, and Fan Liu. Fusing transformer model with temporal features for ecg heartbeat classification. 2019.

[37] Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. 2017.

[38] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. 2017.

[39] Zibin Zhao. Transforming ecg diagnosis: An in-depth review of transformer-based deeplearning models in cardiovascular disease detection, 2023.

Appendix A

Implementation

```
# Transformer Encoder single-lead (TensorFlow)

def get_positional_encoding(seq_length, d_model):
    position = np.arange(seq_length)[:, np.newaxis]
    div_term = np.exp(np.arange(0, d_model, 2) * -(np.log(10000.0) / d_model))
    pe = np.zeros((seq_length, d_model))
    pe[:, 0::2] = np.sin(position * div_term)
    pe[:, 1::2] = np.cos(position * div_term)
    pe = pe[np.newaxis, :]
    return tf.constant(pe, dtype=tf.float32)

def transformer_encoder_block(input, input_shape, num_heads, key_dim, ff_dim, dropout):
    # Multi-Head Attention
    x = keras.layers.MultiHeadAttention(num_heads=num_heads,
    key_dim=key_dim, dropout=dropout, kernel_regularizer=regularizers.l2(0.001))(input, input)
    # Add & Normalize
    res = x + input
    x = keras.layers.LayerNormalization(epsilon=1e-6)(res)
    # Feed-Forward Layer
    x = keras.layers.Flatten(input_shape=input_shape)(x)
    x = keras.layers.Dense(units=ff_dim, activation='relu', kernel_regularizer=regularizers.l2(0.
    x = keras.layers.Dense(input_shape[0] * input_shape[1], kernel_regularizer=regularizers.l2(0.
    x = keras.layers.Reshape(input_shape)(x)
    x = keras.layers.Dropout(rate=dropout)(x)
    # Add & Normalize
    x = x + res
    x = keras.layers.LayerNormalization(epsilon=1e-6)(x)
    return x

def build_encoder(num_classes, input_shape, positional_encoding, num_encoder_blocks, num_heads, k
    inputs = keras.Input(shape=input_shape)
    x = inputs
    if positional_encoding:
        positional_encoding_values = get_positional_encoding(input_shape[0], input_shape[1])
```

```
    x = x + positional_encoding_values
for _ in range(num_encoder_blocks):
    x = transformer_encoder_block(x, input_shape, key_dim, num_heads, ff_dim, dropout)
x = keras.layers.Flatten(input_shape=input_shape)(x)
outputs = keras.layers.Dense(num_classes, activation='sigmoid')(x)
return keras.Model(inputs, outputs)
```

Appendix B

Graphics

t.b.c.