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 [3] [5] [8] [16] [17] [19] [32] [36]. 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] [8] [17] [16] [19] [32] [36]. 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 [16], 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] [18]. 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 accurately 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, such as 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 will be limited to short-term ECGs due to the requirements and to focus on a specific problem (here, short-term single-lead ECGs). An electrocardiogram can be recorded with different devices, i.e. 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, i.e. three leads or up to 12 leads. Special watches [16] 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 [23]. The light is absorbed and reflected by blood vessels and the changes in light absorption are measured to determine heart rate. However, the quality of the sensor is usually not as accurate as a standard ECG machine, 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 [25] to the database provided by Maastricht University, 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-processing filters applied, which will be discussed in section 4.1.3.

Figures 1.1 and 1.2 show graphs for a heart from a physiological perspective and a single normal heartbeat from an ECG, defined as a sinus rhythm. In the literature [2] [18] 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.

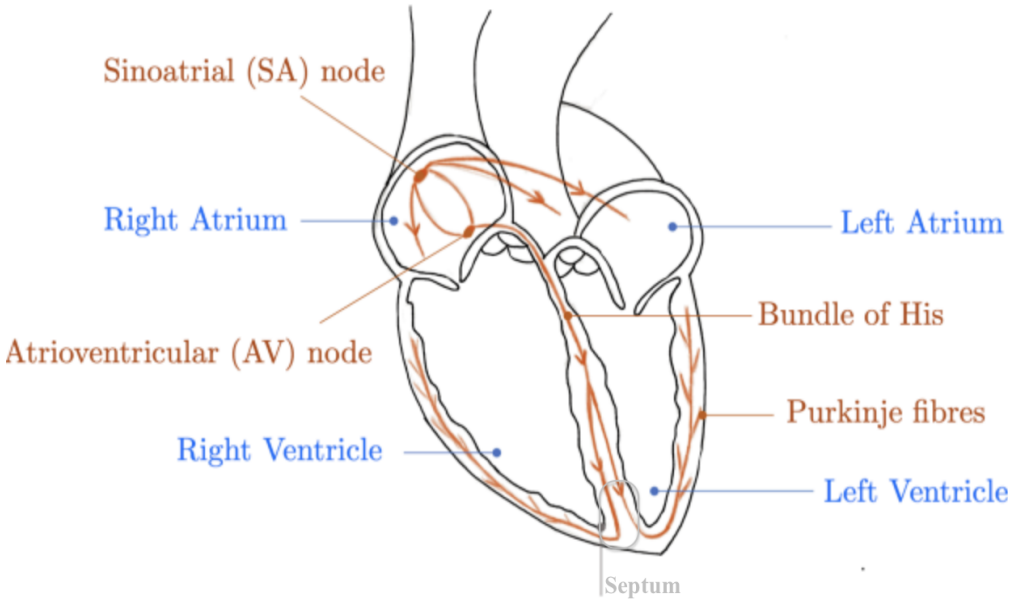

Figure 1.1: Human heart anatomy
https://www.collimator.ai/tutorials/pacemaker-design-and-human-heart-model

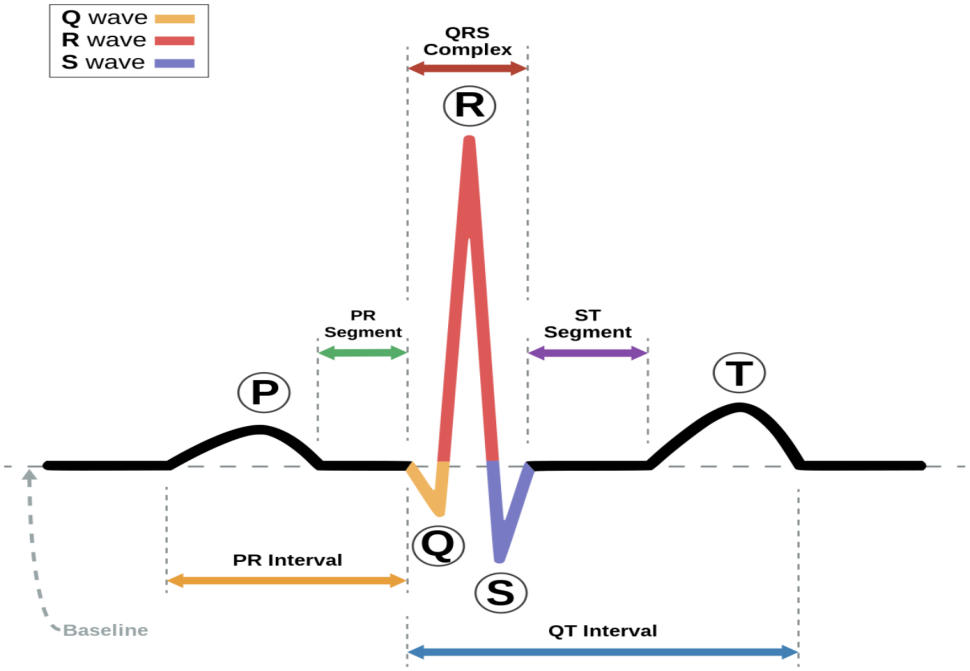

Figure 1.2: Heartbeat rhythm composition
https://en.wikipedia.org/wiki/Electrocardiography

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 repolarisa-

tion 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 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] [18], 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 [27] [26]. 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] [18], 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 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 strongly 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 are 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. This work explores this mechanism to distinguish the harder to detect abnormalities, such as distinguishing between atrial fibrillation and atrial flutter or capturing premature atrial contractions and premature ventricular contractions. Premature atrial and ventricular contractions often occur only once on an ECG. Compared to convolutional filters, transformer models introduce a 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, number of heads and blocks, and further hyperparameter tuning. For the evaluation of the models the Physionet 2021 challenge data [25] 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 below, which will be 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 for transformer based models on ECGs. The second section discusses related work for the Physionet 2021 challenge. The last section discusses related work in the areas of sinus rhythm, atrial fibrillation, atrial flutter, premature atrial contraction and premature ventricular contraction detection.

2.1 Physionet 2021

The winning paper of the Physionet 2021 challenge [20] 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 the 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, the authors use data augmentation to train their model, although the authors do not describe the methodology behind this further. However, the authors have published a follow-up study [21] 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 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 place paper by Han et al. [14] achieves between 0.55 and 0.58 % 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 a loss function. The authors address the goal of achieving high dataset wise 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" [34]. 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 place winning paper of the Physionet 2021 challenge by Wickramasinghe and Athif [31] proposes two convolutional networks with four residual blocks working in parallel and achieving 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 transform (FFT) to the ECGs to obtain the frequency domain. One model then receives the time domain (the pre-processed ECG itself) as input and the other the frequency domain. Each of the models is completed by a pooling layer to reduce complexity, which contains the feature space that is then fed into a common dense layer for the final classification output of the 26 classes.

2.2 Transformer models

Zhao [36] 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] [8] [17] [19] [36]. This is because CNNs have a limited receptive field, which prevents them from learning distant dependencies, and are designed to extract local patterns. Transformers, on the other hand, can learn long-range dependencies through their attention mechanism. The combination of both can capture spatio-temporal information from the ECG signal [36]. Hu et al. [17] 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 [12], 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) [9] was one of the first papers to effectively apply transformer models to computer vision tasks. In vision transformers, an embedding is represented as a subpart/pixel group from an image and the model learns global relations from the whole image by attending at specific parts.

2.3 SR, AF, AFL, PAC and PVC

Wang et al. [30] analyse in their paper an approach, called PVCNet, for the detection of PVC. The authors developed a deep CNN network combined with several dense layers. The input to the model is the full sequence of a single-lead ECG. The authors use the publicly available MIT-BIH database [12] for pre-training and test it on the St. Petersburg INCART database (INCARTDB) [**?**]. 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 [15] the importance of improving the accuracy of assistive ECG devices. They find that the device makes about 18%-24% false positives. 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 and temporal information is extracted. With their approach they 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. [6] focuses on the classification of non PVC and PVC using only the QRS complex. The authors use the MIT-BIH database [12] by extracting all QRS complexes from the long-term ECGs in the MIT-BIH database. In their study they compare several models, i.e. a random forest, LSTM, bidirectional LSTM, ResNet-18, MobileNetv2 and ShuffleNet. ResNet-18 is a deep convolutional network with 18 layers. MobileNetv2 [28] and ShuffleNet [35] are efficient CNN networks designed to work on mobile devices. Ribero et al. show in their experiments [26] and [27] 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 data collected from various cardiologists and hospitals. However, in their paper [27], the authors state that the performance of the model trained only on the Physionet 2021 databases is significantly lower than that of the model trained only on their private dataset. Wang [29] 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) [11], 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 [12] database for the binary classification problem.

Chapter 3

Approach

3.1 Mathematical background: Transformer

Transformer models [1] have the capability of processing long-range sequential data a lot better using a mechanism called self attention. In recent years, transformer models have gained significant popularity, because their model and training design can be much easier accelerated on a graphic card (GPU), which is due to the capability of parallel weight updates and shared weights across the attention (query, key and value) matrices. This leads to a significant decrease in training time and hardware costs, while maintaining comparable or even better performance, compared to the former long short term memory (LSTM) and recurrent neural network (RNN) model architectures, which have been widely used until then for sequential data. LSTMs and RNNs need to process data sequentially through their units and do not scale very well for large training applications. In addition these models can suffer from long-range dependencies, caused by vanishing gradients during the backpropagation. In the following the architecture and training design of the transformer model for the original application of language tasks is roughly explained. Although, this might be a bit off topic, the general idea is to explain the origin key idea and highlight task-related considerations for a transformer architecture and training design from a language based model compared to a transformer trained on ECG signal data.
The underlying transformer model from the original paper "Attention Is All You Need"[1] is composed 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 has been used for language translation by mapping the encoded tokens (words) from one sequence to the translated sequence. Many adopted this architecture and use parts of it for a wide range of down-stream tasks, i.e. for classification tasks, such as BERT (bidirectional encoder representation from transformers) [7] or for prediction/sequence completion tasks, such as GPT (generative pre-trained transformer model) [4], which uses only the decoder block. Transformer models usually split the input sequence into subspaces as vectors, so called embeddings. In language oriented tasks an embedding represents a word (more precisely a subword), in this thesis an embedding represents a subpart in a 1-lead ECG signal or a single heartbeat. Embeddings represent the input tokens in language oriented tasks. The input embeddings represent the first layer of a transformer model. Researchers came up with different strategies to alter the model, i.e. BERT [7], which was designed for text classification and Q&A. In BERT the input and output layers are adjusted for several fine-tuning tasks, i.e. for classification tasks a special classification token [CLS] has been added to BERT's input and output layer or for Q&A tasks, BERT adds beside positional encoding, segment information to the embeddings and an additional separation token

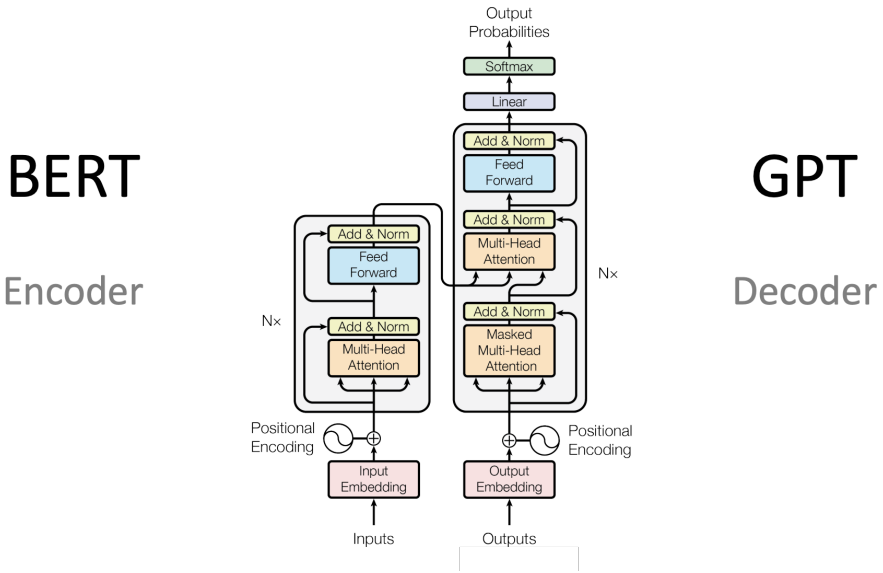

Figure 3.1: Transformer architecture

[SEP], thus a pair of sentences can be distinguished and inputted to BERT. BERT makes use of the encoder block to learn a contextualized representation from the input sequence. As a result BERT outputs the same number of embeddings as received in the input layer, but the outputted embeddings are enriched with context information from the embeddings in the attention block through the attention mechanism. However, BERT uses at the same time an [CLS] output weight for classification tasks. The training process of BERT was done by two tasks, masked language modeling (MLM) and next sentence prediction (NSP) [7]. Both, MLM and NSP, masks words/embeddings in the input sequence with pre-defined masking embeddings to predict them based on the other embeddings. The attention block can be compared to a convolution in CNN's, but are designed as weighted, global attention filters, which forces the model to attend specific areas while merging embeddings. The attention block will be detailed discussed in the next section. In the original BERT paper 12 encoder blocks have been stacked [7]), where each block adds incrementally more context information from all the other embeddings to each outputted embedding. BERT has been further adjusted to other tasks, for example semantic search, proposed in SBERT [24], where a pooling layer is added on top of BERT, which denses the output embeddings into a single embedding, that can represent the entire input text and be used for fast semantic search. These embeddings can then be compared by dot product or cosine distances in vector databases and used for fast semantic similarity search. Semantic textual similarity, another concept to adjust BERT, performs cross-attention on a pair of embeddings in the input sequence and outputs directly a similarity score [24]. Compared does GPT use only the decoder block, which is based on a similar attention mechanism. Although, the attention mechanism in the decoder block focus the attention only on past states (not bidirectional) and incrementally unmask future embeddings, which have to be predicted. For this a sliding window is used. Moreover, GPT uses different sampling and penalty techniques, when generating output. GPT's performance has been achieved by huge amount of training data and the model alignment/behaviour to specific prompt, meaning that transformer models need a lot of data. In GPT-3 96 decoder blocks have been stacked [4]. BERT and GPT have both limited context

lengths, which means that only fixed size sequences can be inputted and otherwise these are padded by special tokens or the sequence will be truncated. Moreover, BERT and GPT both uses special designed tokenizers for the specific use cases and languages. Tokenizers are main difference from the transformer model applied in this thesis, since the transformer model for signal data does not use any signal mapping, compared to tokenizers that are used for transformer models trained on language. Tokenizers are vocabularies, representing the words that are known or can be outputted by the model. Tokenizers are fix-sized, since the model architecture is also given. A tokenizer maps an input text into ID's, where each ID represents a subword or character in the input text. These ID's are unique for each subword and then map to an embedding matrix in the model, where the corresponding embeddings are feeded as inputs to the model as the model receives the ID's from the tokenizer. The embedding matrix, representing all embeddings for each token is part of the model, while the tokenizer is not part of the model and a separate module, representing a vocabulary, lookup table for ID's and indexes for the embedding matrix in the model. The embedding matrix is randomly initialized and the embeddings will be updated during the training procedure. The tokenizer split the input sequence from texts into words, to sub-words, to ID's (numbers/tokens), where each token maps to a specific entry in the the embedding matrix in the model to feed in the correct embeddings into the model. During the output/generation phase this process is vice versa, using a softmax activation function to activate the most probable output tokens and mapping them. The tokenizers include all known words of the model and also special tokens, i.e. separation, padding and out-of-vocabulary (representing unknown words) tokens. The tokenizer can drastically reduce the models embedding matrix and thus the required training time, as much more training would be needed to reach the desired adjustments for a much larger embedding matrix containing every out-written word. An example would be that a special token in a common tokenizer is for example the suffix -ly, so the model does not to know every adverb. In BERT's case the tokenizer has fixed sized vocabulary of $\sim$30000 sub-words. In this thesis no tokenizer and mapping procedure of the ECG signal data is used. Moreover, no embeddings matrix is updated during the training procedure, since each signal is individual and the parts do not correspond to unique words. Only the attention query, key and value matrices will be trained. Although this could be considered for future work to cluster related signals or beats.

The self attention block calculates the relation among each embedding from the input sequence and it relative importance to the other embeddings from the input sequence, yielding attention scores [1]. Other than convolutional filters in CNN's, this introduces weights to the kernels to merge the input weight-based. Convolutional filters or kernels in CNN's are locally, while this trained attention block and it's entries can be thought as global kernels. In the original paper and model this process is called the self attention mechanism [1], which is the overall strength behind the transformer model.
While the transformer encoder block takes into account the bidirectional contextual representation from the entire ECG signal to capture temporal dependencies from the past and the future, the decoder block sequentially processes the signal from left to right and only focuses the attention on past states by masking future states. However, [33] investigates the application of transformer based models for time series forecasting and find that simple linear model outperform the transformer model in several experiments. They argue that transformers may not be as effective due to their permutation invariant nature, which can result in a loss of temporal information. The objective of this work is to investigate and compare the performance of different encoder based transformer architectures with 1D-CNN's and none deep learning models, for the classification of arrhythmia diseases. The work will compare different encoding strategies, by using extracted conventional features for the simpler models, i.e. from the RR intervals, or by

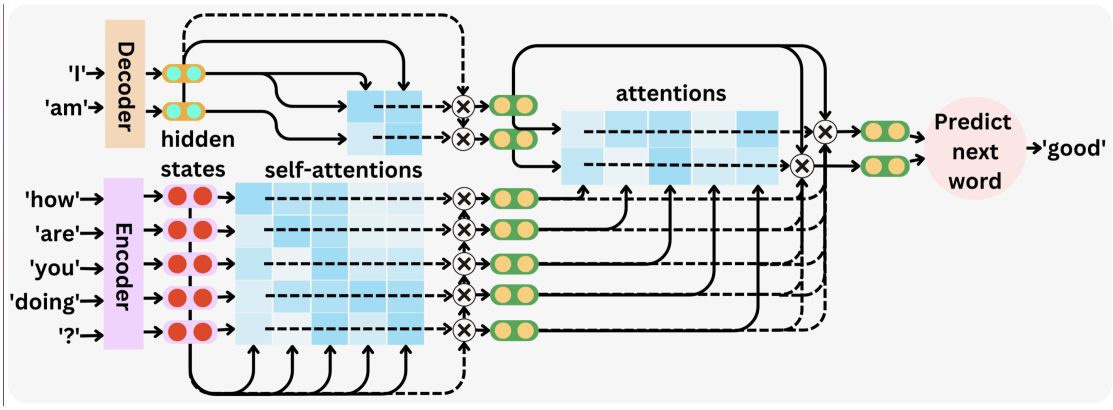

Figure 3.2: Encoder and decoder interaction

using the entire ECG as input for the more complex deep learning models, which learn features by themselves 1.1.2.

3.1.1 Self attention

In more detail the attention mechanism works as follows:

The attention block uses a query, key and value matrix (represented by the three arrows, which go into the attention block in figure 3.1). The query, key and value matrices will be trained. Each embedding in the input sequence is multiplied with the query, key and value matrix, one entry/vector/column in each of the three matrices is multiplied with one embedding from the input sequence, thus the matrices have the same dimension as the number of embeddings in the input sequence times the dimension/length of each embedding. This yields three new embeddings for each embedding from the input sequence, one query embedding, one key embedding and one value embedding. In the next step the distance (dot product) from each query embedding to each key embedding is calculated, yielding the quadratic number of similarity scores. The obtained values are used as scaling weights (attentions scores). The attention scores are normalized for further calculation. Similar to a softmax function, the weights are used to enrich each value embedding with weighted sums from the other value embeddings, by multiplying/scaling the attention scores (from the current query embedding) with all the value embedddings and add the scaled value embeddings to each value embeddings (based on the current query embedding and attention scores).

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

Figure 3.3: Formula for attention scores

First, for each embedding from the input sequence the corresponding query, key and value embeddings will be calculated by multiplying the embeddings with the corresponding entry/column in the query, key and value matrices (see W^Q, W^K and W^V yielding the Query, Key and Value

matrices in 3.4, where each entry in each of the matrices represents a new transformed query, key and value embedding for each of the embeddings in the input sequence). The advantage is that the weights in W^Q, W^K and W^V are shared during this entire forward pass and do not depend on past states. Next the attention mechanism iteratively focus on one query embedding at a time per to calculate the similarity to all other key embeddings (and itself, called self attention) from the input sequence using the dot product. This leads similarity weights to the quadratic number of embeddings in the input sequence (each query embedding separately to all other key embeddings: the query embeddings to the power of key embeddings). The similarity weights are represented in weight matrix A in 3.4). In the next step the weights are normalized by a softmax function, yielding scaling weights (attention weights) for each embedding (see weight matrix B in 3.4). Finally, the scaling weights are used to weight-based merge each value embedding with each other value embedding using again the dot product (yielding matrix Z in 3.4). As a result this leads to adjusted value embeddings, which is the calculated value embedding plus the weighted sum from all the other (and itself) value embeddings from the input sequence. The scaling weights (matrix B in 3.4) represent how much each value embedding should attend/contribute to each value embedding. The most influence to the value embedding during this weight-based merging process will be value embedding itself, as it is most similar to itself compared to the other embeddings, thus the value embedding will keep most of its own information and is only partly weight-based influenced by the other value embeddings. Notice that the attention-block has a limited context size by sing a sliding window ("molecular"

Compared to a convolution this attention process merges all embeddings global, based on the corresponding scaling weights obtained from query and key embeddings similarities, which are used to scale the value embeddings during the value embedding merging process (from the perspective of each input embedding separately) and not only incrementally convolve the input on their local neighbourhood as it is the case in a convolutional Network.

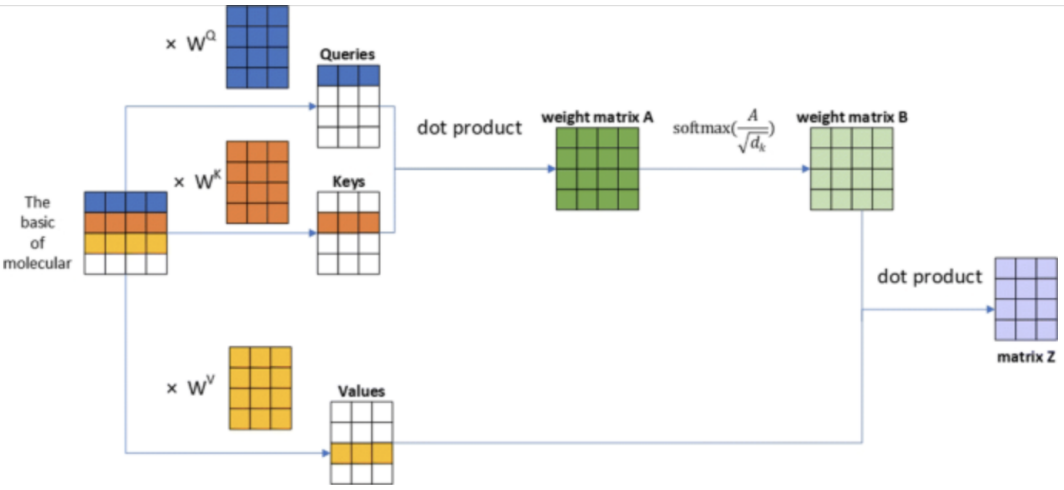

Figure 3.4: Stepwise calculation of the self-attention module

3.1.2 Multi-head attention

The original paper [1] introduces on top a strategy called multi-head attention, which splits up each embedding into N embeddings (heads), to train N attention matrices at the same time, by focusing on each input embedding from another angle/perspective, which enables to learn ambiguity. In the original paper 8 heads has been used [1]. This means an embedding (representation of a word/token), for example 512-dimensional long embeddings, would be splitted up into 8 subparts, yielding 8 x 64 dimensional embeddings for each embedding, where 8 attention-blocks are trained at the same time on the different subspaces from the embedding. Finally, the subembeddings are then concatenated again after the attention-block.

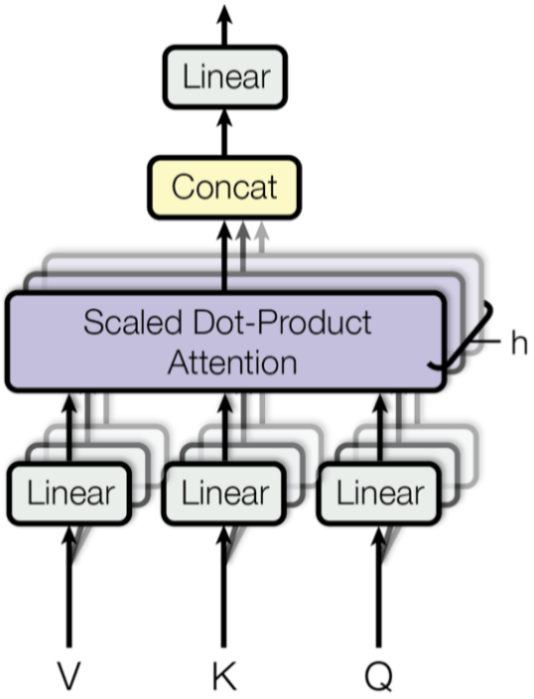

Figure 3.5: Multi-head attention

3.1.3 Positional encoding

Moreover, the standard transformer model adds positional encoding on top of the embeddings to add positional/temporal information of the order from to the input embedding sequence, which will be loosing otherwise through the fuzzy process of the attention block, by adding a residual connection before and after the attention block, which adds the positional encoded embeddings from the input sequence before the self-attention block with the the contextualized embeddings after the self-attention block. The original paper proposed a positional encoding technique, which uses ... function ... t.b.c

3.2 Mathematical background: Convolutional network

As a second comparison model, a convolutional network is trained using Tensorflow. Convolutional networks are successful and frequently applied deep learning based models to the classification of ECG signals (including the models in the three winning papers from the Physionet 2021 challenge) [20] [14] [31] and [15] [6] [30]. Initially designed for image classification these models can also be applied to 1D 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

For baseline comparison, a feature-based model is trained using 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 R peak locations and other fiducial points in the ECG, e.g. P, Q, S and T peaks and their wave onset/offset points. For the peak localisation the package has one of the three methods implemented: "pantompkins" [22], "hamilton" [13] or "elgendi" [10]. As the feature-based model is just a baseline comparison, only the "pantompkins" method has been used in the experiments. These methods are also the standard implemented algorithms for the fiducial points localisation in the Neurokit2 package. The Neurokit2 package is a more maintained library to work with ECG data, but which is due to similar methods and limitations 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 4.4.

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

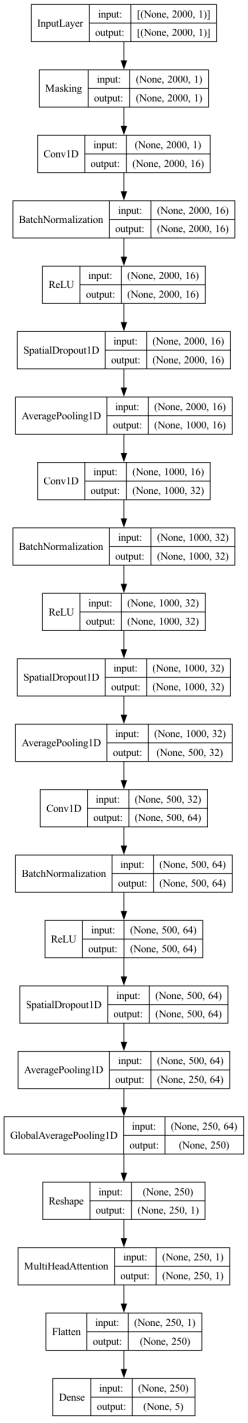

Figure 3.6: Physionet 2021 class distribution - SR, AF, AFL, PAC and PVC

3.3.3 Residual CNN

3.3.4 Ensembled models

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) [25]. 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 [14].

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.

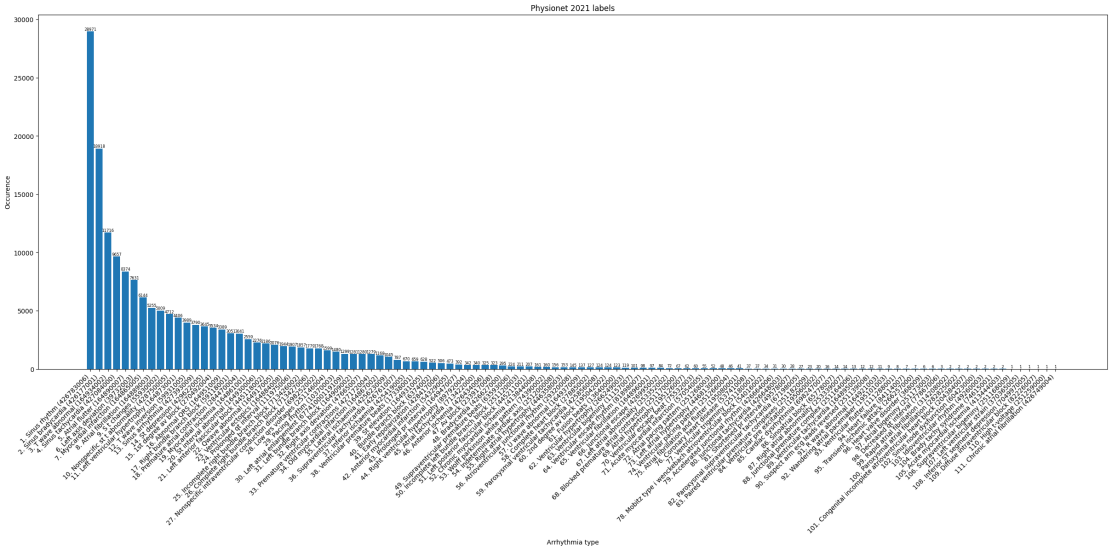

Figure 4.1: Physionet 2021 all classes distribution

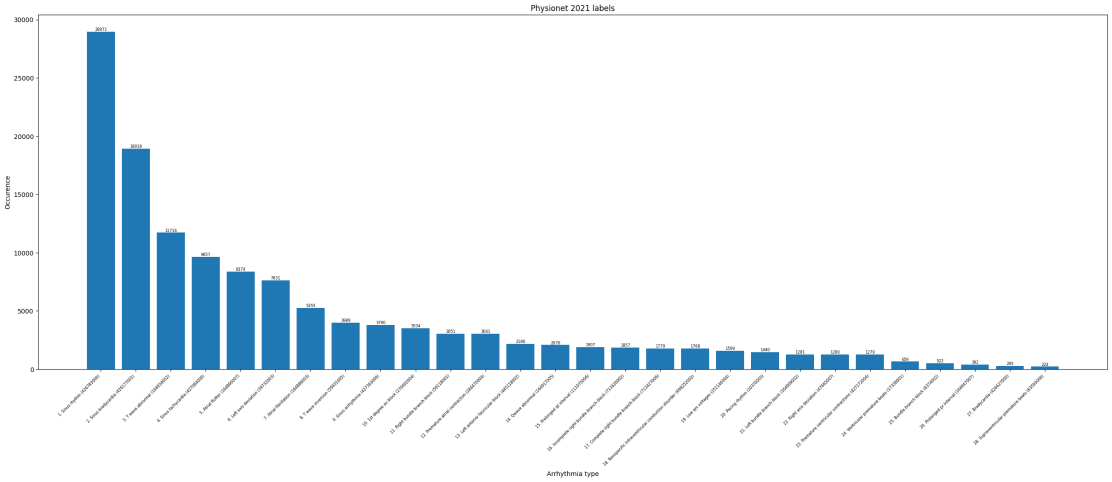

Figure 4.2: Physionet 2021 scored classes distribution

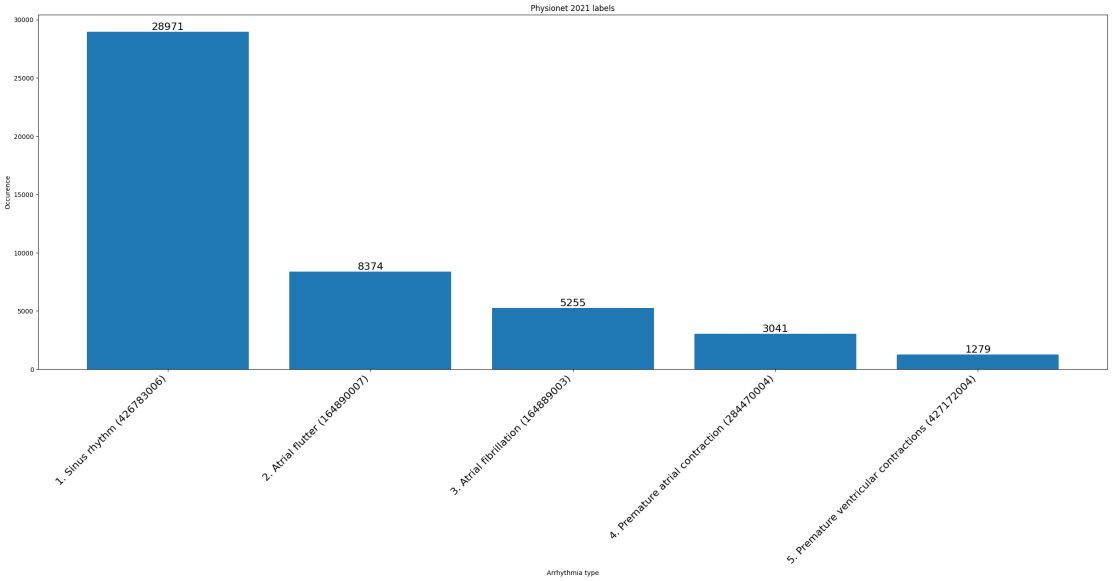

Figure 4.3: Physionet 2021 class distribution - SR, AF, AFL, PAC and PVC

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.

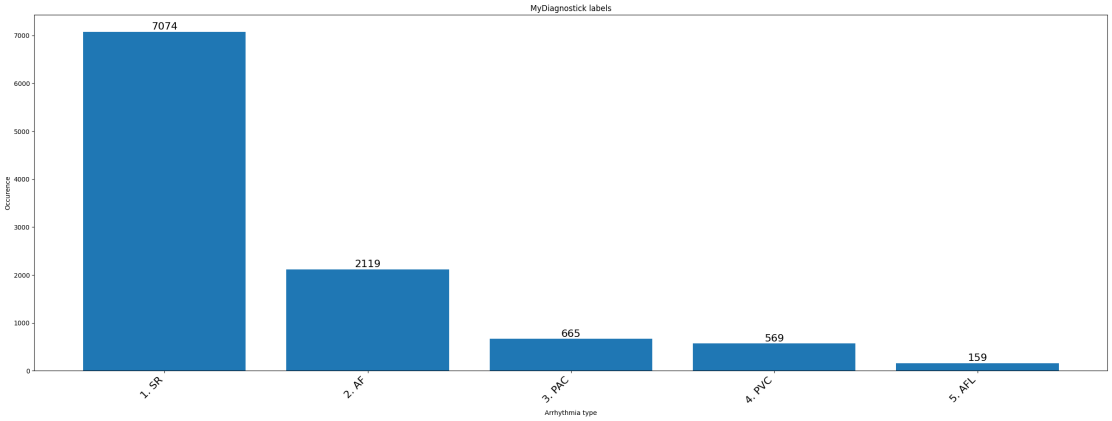

Figure 4.4: MyDiagnostick class distribution - SR, AF, AFL, PAC and PVC

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 a 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

4.3 Results

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

4.4 Discussion

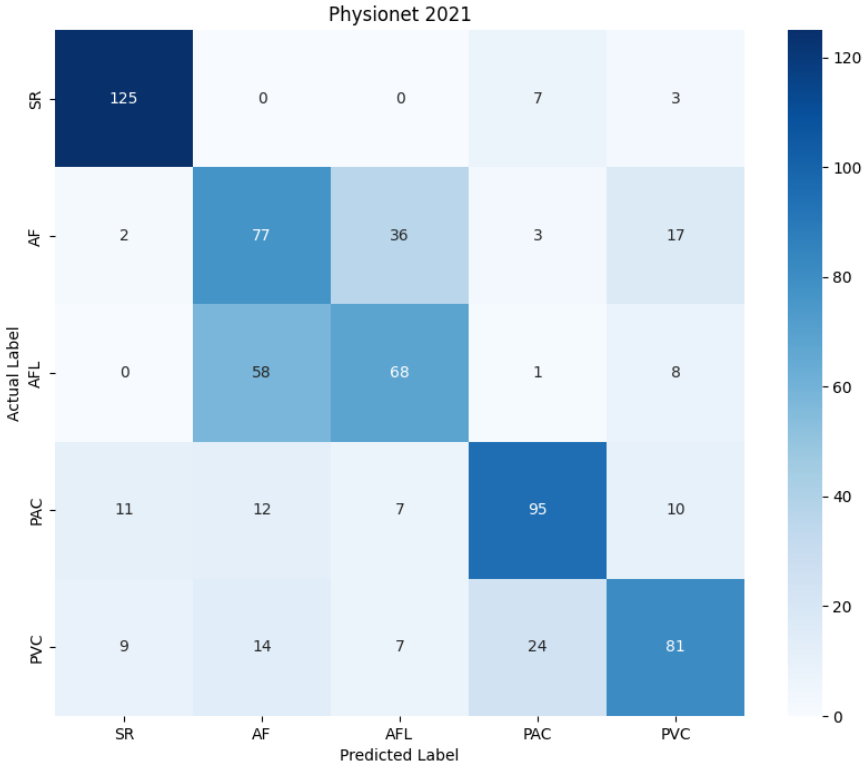

Figure 4.5: Confusion matrix - Residual CNN: SR, AF, AFL, PAC and PVC

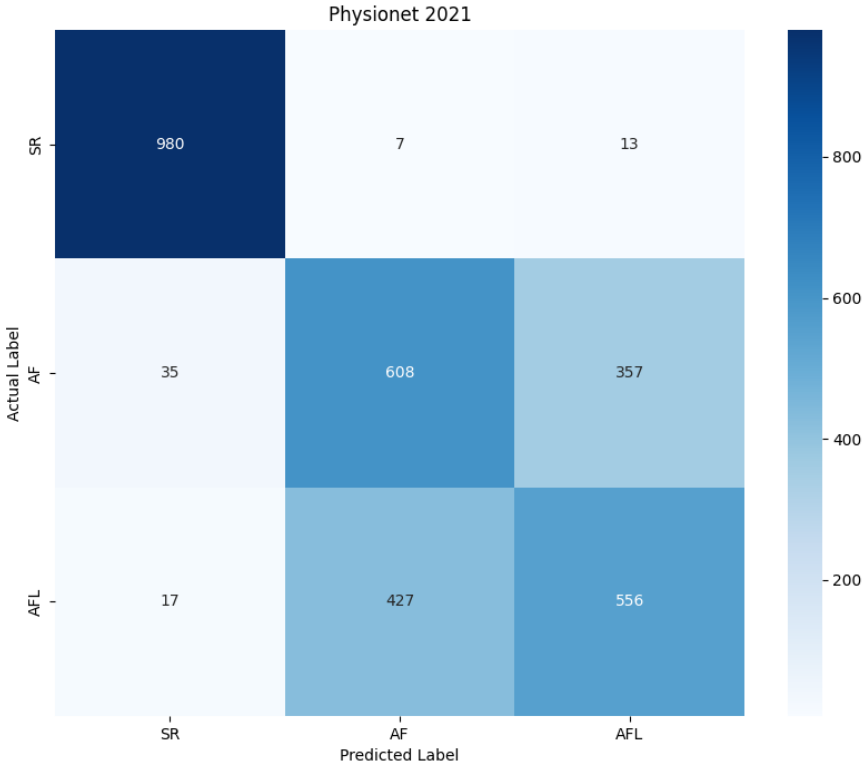

Figure 4.6: Confusion matrix - Residual CNN: SR, AF and AFL

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

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

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

[9] 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.

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

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

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

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

[14] 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.

[15] 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.

[16] 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.

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

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

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

[20] 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.

[21] 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.

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

[23] 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.

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

[25] 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.

[26] 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.

[27] 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.

[28] 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.

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

[30] 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.

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

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

[33] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting?, 2022.

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

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

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

Appendix A

Implementation

t.b.c.

Appendix B

Graphics

t.b.c.