Master Thesis

Analysing various CNNs and attention mechanisms for single-lead multi-label ECG classification

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. Pietro Bonizzi
Prof Dr. Stef Zeemering
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] [21] [25] [44] [47]. 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] [21] [18] [25] [44] [47]. Multi-lead ECGs provide a view of the heart's electrical activity from different angles and can localise abnormalities more reliable. 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] [22]. 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 [29]. 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 [31] 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] [22] 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.

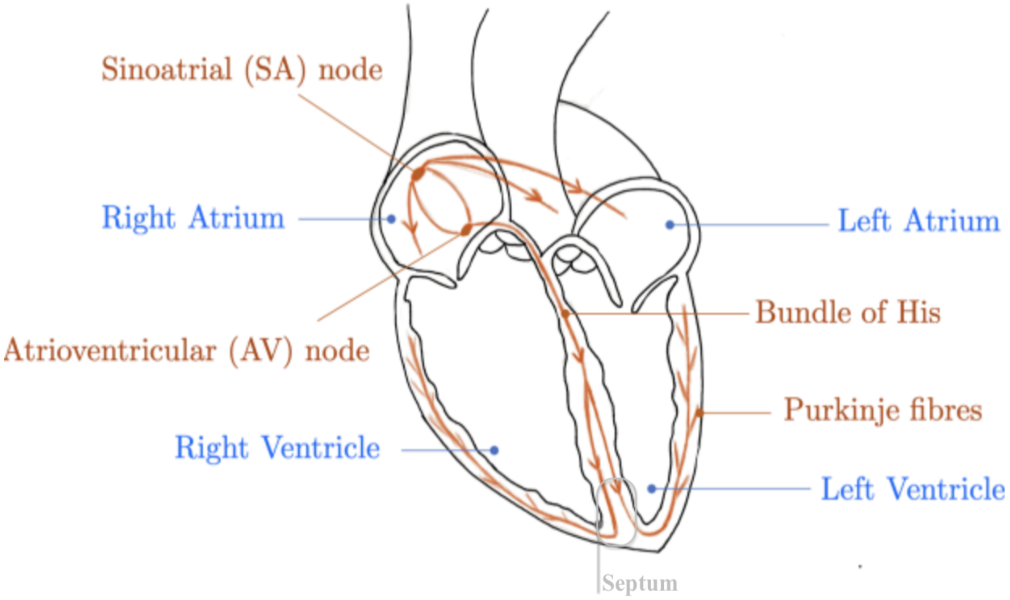

Figure 1.1: Human heart anatomy
Source: Collimator

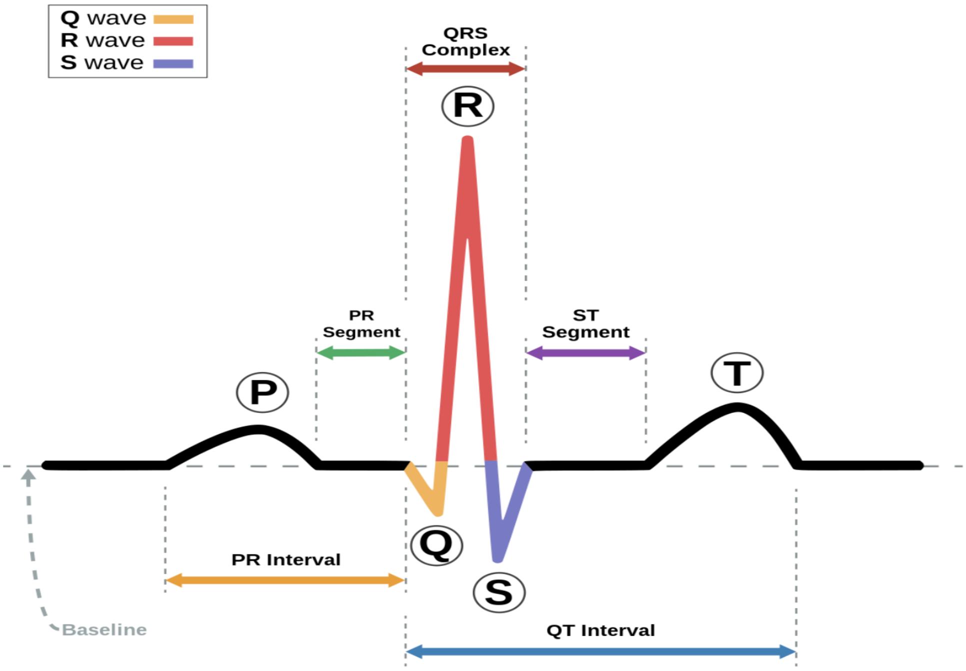

Figure 1.2: Heartbeat rhythm composition
Source: Wikipedia

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] [22], 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 [33] [32]. 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] [22], 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 [31] 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. Can an ensemble Transformer model and Convolutional Network effectively capture spatio-temporal information and improve accuracy?

3. Which model performs best at discriminating SR, AF, AFL, PAC and PVC on both datasets?

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

Several paper in the Physionet 2021 challenge incorporate an attention-based mechanism. To give an overview the four first ranked papers are discussed, from which the first and fourth ranked paper integrate an attention mechanism. The fourth ranked paper uses several different attention-based strategies and is discussed in more detail. The first ranked model of the Physionet 2021 challenge [26] by Nejedly et al. proposes a deep residual CNN network, which uses multi-head attention. The model is designed for 12-lead ECG classification, while for fewer lead configurations the unused leads are padded with zeros. Their approach achieves about 58% on the 2, 3, 4, 6 and 12-lead tasks. The model uses large convolutional filters, i.e. 15x15 on the first convolutional layer and 9x9 on subsequent convolutional layers. The model integrates a single multi-head attention block, to capture temporal relations. However, in a follow-up study [27] the authors find that the multi-head attention block does not improve classification performance. Based on their experiments, the authors find that without the multi-head attention block, the model achieves an overall accuracy of 58% and with the multi-head attention module only 57% on the official Physionet 2021 challenge test data. Furthermore, the authors propose a loss function that incorporates the challenge evaluation metrics. The second reanked paper by Han et al. [15] achieves between 55 and 58 % accuracy. In this approach the authors use a deep convolutional network combined with demographic features, e.g. age and gender. The features are integrated in the output dense layer. The authors highlight that the model achieves high generalisation performance by using a "leave-one-dataset-out-cross-validation" strategy. Moreover, their approach use a data augmentation method called "Mixup" [45], which makes the decision boundaries of the model smoother through a regularisation technique that mixes two input samples with their features and labels based on a coefficient. The third ranked paper by Wickramasinghe and Athif [43] proposes two deep Convolutional Networks with four residual blocks, which work in parallel and achieves an accuracy of 51%-55% on the final test set. The authors apply standard pre-processing steps such as normalisation, resampling and zero padding. One Convolutional Network receives the raw ECG signal with temporal information as input, while the second Convolutional Network applies a Fast Fourier Transformation (FFT) on the ECGs to capture features from the frequency domain. Both models are combined by a dense and pooling layer. The fourth ranked paper by Srivastava et al. [37] proposes a residual Convolutional Network, which incorporates several convolution- and attention-based methods. The methods include inception proposed by Szegedy et al. in [38], channel-attention proposed by Hu et al. in the squeeze and excitation network [20] and attention-pooling proposed by Ilse et al. in [23]. Back in 2014 the inception mechanism set new state of the art performance on the ImageNet challenge [35], while using much fewer trainable parameters. The idea of inception is to use several convolutional filters in parallel in the same convolutional layer, e.g. in the original paper 1x1, 3x3 and 5x5, here 1-d convolutions with 1x3, 1x4 and 1x5. The outputted stacks of feature maps are then concatenated into one feature map block for further processing. Although, the original paper apply 1x1 convolutions in the inception module, the authors from the challenge paper [37] do not apply these 1x1 convolutions. The idea behind 1x1 convolution is that it reduces dimensionality using a trainable convolution that applies on channel depth. Unlike pooling, which takes the minimum, maximum or average values within the same channel, 1x1 convolution applies dimension reduction across all channels in depth for which it uses trainable parameters. The expression of 1x1 convolution might be a bit misleading, because more precisely it applies a "number of channels"x1 convolution in which each value of all feature maps are weighted-based reduced into one channel. Therefore, the overall strength in inception is extracting and combining features from different kernel sizes, while keeping the dimensional overhead low using 1x1 kernels. Although, the term "attention" is not directly used in the inception paper [38], the 1x1 convolution could be compared to attention by learning a set of linear weights that merge channel information, while channel-attention, which is described next, incorporates non-linear functions like relu and sigmoid to weight channels as attention mechanism. Instead of 1x1 convolutions, the authors adopt channel-attention in their architecture. Channel-attention is implemented through a network architecture, called squeeze and excitation network by Hu et al. [20]. First, a squeeze and excitation network squeezes channel information by global average pooling, which is then fed into a two hidden layer feed-forward network using relu (first) and sigmoid (second) as activation functions. The network is intended to learn inter-dependency among channels. The first hidden layer uses a dense ratio to the number of channels, which has been emperically determined in the original paper [20], e.g. r=8, r=16 or r=32. For example, an input vector with 256 averaged values obtained from previous global average pooling operation with 256 channels, the first hidden-layer size for r=8 would be 32. The dimensionality is then increased in the second hidden layer to the number of channels again using sigmoid as activation function for scaling. The outputted values are used as channel-attention weights to scale the inputted channels, which are propagated throughout the network beside the squeeze and excitation block (excitation operation). The authors alter this method to channel self-attention. Instead of using global average pooling the authors extract 32 features from each channel and calculate attention within each channel based on the feature vector. Interrelation among channels is captured through sharing weights across channels within the squeeze and excitation module. Furthermore, the authors extend the model in the output with a technique called attention-pooling [23] and alter it multi-head attention, which pools the final feature vectors into a feature space of size NxL, where N is the number of predicted classes and L the feature aggregation. The inputted feature vectors are subdivided into N segments. The attention block uses trainable weights U and W that linearly project the feature vectors, apply matrix multiplication and then softmax to scale the weights. The scaling values are multiplied with the inputted feature vector. However, the paper [37] does not state, whether the inputted feature vectors are the feature maps from the previous channel-attention layer.

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

Choi et al. [6] propose an encoder-based Transformer model for ECG arrhythmia classification, called ECGBERT. Based on the architecture of BERT [8], the model is fine-tuned on multiple down-stream tasks, e.g. atrial fibrillation classification (binary), heart-beat classification (normal, unknown, supraventricular ectopic, ventricular ectopic and fusion heartbeats), user verification by using the ECGs as biometric authentication to predict patient IDs and sleep apnea detection. For their model the authors create an own wave segment vocabular to tokenize each ECG and apply then similar to BERT, Masked Language Modeling (MLM), to pre-train the model. In a preparation step, ECG signals are preprocessed (normalized, filtered etc.) and splitted by fiducial points into wave segments. Time-frequency analysis and the Hamilton algorithm [28] are applied to clean the ECG signals and extract onset and offset points. The obtained wave segments are used to cluster the waves using K-mean and Dynamic Time Warping. For this, the authors train four classifiers: P, QRS, T and background waves. 70 clusters are obtained with 12 P, 19 QRS, 14 T and 25 background (e.g. PR or ST intervals) wave clusters. Next, each segment is classified by the corresponding classifier, assigned to a wave cluster and the classified wave segments are mapped to pre-defined tokens. The authors do not describe how the tokens are encoded, an assumption could be that these are random initialized embeddings. In addition, CNN features are extracted from the raw ECG using an U-Net [34]. The authors reason that the model can only learn high-level wave context relations from the ECG tokens, but CNN features provide refined pattern information and therefore are crucial for extracting fine-graned features. The tokens are combined with the CNN embeddings and positional information is added, likewise in the original Transformer paper [1]. Based on this, the authors create training samples from the ECGs that form sentences of wave segments, where each sentence contains 1-8 consecutive heartbeats and a heartbeat is composed of several wave segments. The constructed ECG wave sentences are then inputted to the Transformer-encoder module, where several sequences can be inputted, which are splitted by a seperation "[SEP]" token. The model is pre-trained using 15% masking (similar to Masked Language Modeling in LLMs). For pre-training of the Transformer the authors use the MIT-BIH database (AFIB/arrhythmia) [13] and the Apnea-ECG database from Physionet. Based on the pre-trained model weights the authors use the model to fine-tune it on several down-stream tasks by adapting the output layer to the corresponding tasks. The model achieves for binary classification of Atrial Fibrillation an accuracy score of 97% and for beat classification about 86%. Zhao [47] lists in a review from 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 Transformer-encoder block with a Convolutional Network [3]. [5] [9] [21] [25] [47]. The authors reason that CNNs have a limited receptive field, which prevents them from learning distant dependencies and extract local patterns. On the other hand, Transformers learn long-range dependencies through their attention mechanism. The combination of both capture spatio-temporal information from the ECG signal [47]. Hu et al. [21] 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. [42] 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 [36] and ShuffleNet [46] are efficient CNN networks designed to work on mobile devices. Ribero et al. show in their work [32] and [33] 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 [33] 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 of the Physionet data. Wang [41] 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 utilized and modified approaches developed for this thesis. It is divided into three sections. The first section discusses the mathematical background of Transformer models. It highlights the architectural differences between a Transformer model trained on language and applied to ECG signals. The second section section gives a brief overview from the main components of a Convolutional Network. The third section presents the modified approaches that are used to address the problem of multi-label ECG classification into 26 different arrhythmia classes, here a feature-based classifier, a Transformer model, a Convolutional Network and several ensemble models.

3.1 Mathematical background about Transformer models

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.

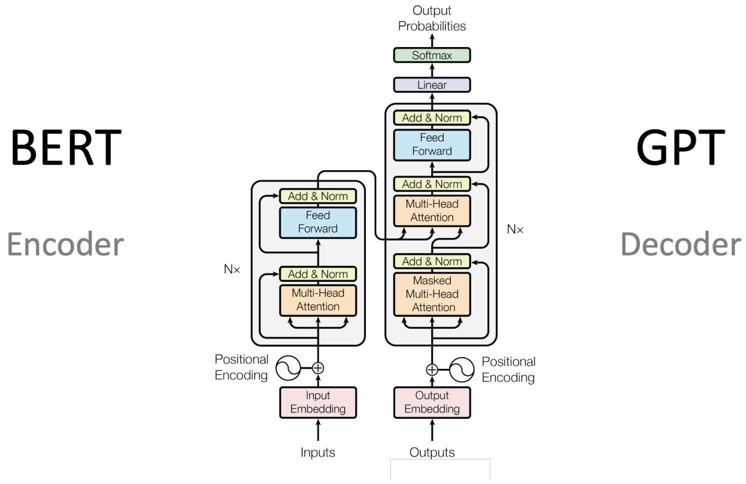

Figure 3.1: Transformer architecture
Source: ”Attention Is All You Need” [1]

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 projects and merges each feature with surrounding features from the input sequence. However, the objective of the Transformer model is to learn relations within the input based on weighted attention scores, unlike the Convolutional Network, which aims to extract invariant features from local inputs and feature groups. That means, depending on the kernel size, a convolutional filter applies only to local input and feature groups, while the attention block applies to the entire input sequence as whole. The attention block computes query, key and value pairs by considering each input element with each other input element individually. During training, the attention block adapts its attention weights to different arrangements in the training data, i.e. as it sees different variations in the training data it can extract relationships and learn meaningful patters from various data arrangements, by forwarding the most important relations and features. The attention mechanism is in more detail discussed in section 3.1.2. Many researchers have adopted the original Transformer architecture and use parts 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 utilises and implements only the encoder block of the Transformer model. The reason is that the encoder block considers all inputted embeddings (bidirectional), whereas the decoder is designed for prediction tasks and focuses its attention on the output of the encoder block, also referred as cross-attention, and on its own past states (unidirectional) by masking future states. This is achieved by adding masking to the embeddings in the decoder block. 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-based Transformer 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 different as in this work and is usually 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 the Transformer model. In this work, simpler signal pre-processing steps are applied before passing the raw ECG data to the Transformer encoder block. This includes splitting the signal into fixed-size segments or features maps / channels extracted by a Convolutional Network. This will be analysed and discussed in the evaluation 4.2 and discussion 4.3. The input size of a Transformer model needs to be fixed-size, also known as the context window of the model. In this work, this is either a 10 or 60, depending on the investigated experiments, seconds long single-lead ECG signal. The ECG embedding features are fed directly into the encoder block, whereas in language-based Transformer models an additional trainable embedding matrix represents first layer of the model to the encoder block.

3.1.1 Positional encoding

The original Transformer model adds positional information to each embedding before these are passed into the attention block. The positional encoding layer is connected with the output of the attention block via a residual connection. This ensures that the 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:

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

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

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

3.1.2 Attention mechanism

The input to the attention layer (Transforemr encoder block) is a sequence of length N, consisting of N positional encoded embeddings (features) with dimension d, e.g. a tensor of the shape $N_{sequence\,length} \cdot d_{embedding\,dimension}$. The attention layer uses three trainable query, key and value matrices for projecting the input sequence into three new feature spaces, denoted by the matrices W^{Q}, W^{K} and W^{V}. The multiplication of the input with each weight matrices W^{Q}, W^{K} and W^{V}, yields the matrices Q, K and V that go into the attention layer, which are represented by the three arrows in 3.1. An important advantage is that the weights W^{Q}, W^{K} and W^{V} can be computed in parallel during the forward and backward pass and do not depend on sequential 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 with K^{T}, which gives an unscaled attention weight matrix A. Since every projected embedding in Q is multiplied with every transposed projected embedding in K^{T}, the shape of the attention weight matrix is $N_{sequence\,length} \cdot N_{sequence\,length}$ (the quadratic number of the inputted sequence). A disadvantage is that the trainable projection weights W^{Q}, W^{K} and W^{V} scale quadratic with the length of the input sequence. The obtained attention matrix A represents how much each embedding relate to each embedding in the same sequence (self-attention), which is learned through W^{Q}, W^{K} and W^{V} and which is utilized to scale the embeddings in the projected value matrix V by taking into account relations among the embeddings. The attention scores in A

are divided by a normalising constant $\sqrt{d_k}$ (the row dimension of W^{K}, which is of shape $d_k \cdot N_{sequence\,length}$, more precisely $h_{heads} \cdot d_k \cdot N_{sequence\,length}$ - but which is discussed in more detail in the next section about Multi-head Attention 3.1.3), which helps the model to reduce the variance of the computed relational values from Q and K and stables training. Next, the attention weight matrix A is scaled by a softmax function to obtain softmax-normalised and interpretable weights. The softmax-normalised weights are the final attention weights of A that will be used to scale the value matrix v by matrix multiplication from the obtained attention matrix A. This operation forces the model to attend and propagate specific embeddings more than others in the network. A linear matrix 3.3 is followed after the calculation with V. The linear matrix, which should not be interchanged with the additional feed-forward layer as part of the Transformer encoder block, is a trainable linear weight matrix that is used to weight-based concatenate the heads, which will be discussed in the next section 3.1.3. Usually, the outputted shape of the sequence and embeddings from the Transformer encoder block should have the same shape as the inputted sequence and embeddings to the Transformer encoder block, e.g. $N_{sequence\,length} \cdot d_{embedding\,dimension}$. Depending on the dimension configuration of W^{Q}, W^{K} and W^{V}, the initial shape might be restored by this linear matrix operation, which is also discussed in more detail in the next section 3.1.3. The masking operation (optional) 3.2 is skipped here, because it is only used in the decoder block and rather suitable for temporal forecasting tasks. In short, the masking operation adds a triangular matrix masking to the attention matrix A so that the processed attention scores depend only on unmasked and past embeddings. The output of the encoder block is added with the input embeddings by a residual connection, which is an important step. This ensures that the initial feature and it's temporal information is propagated throughout the entire network, when stacking multiple Transformer encoder blocks. In addition, layer normalisation is applied to stabilise the weights during 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 a ReLU activation function is used [1]. For comparison, GPT-3 uses a feed-forward layer with a hidden-layer size of 24. The output of the feed-forward layer is then reconnected by a second residual connection and again normalised. Since the output of the entire encoder block should have the same shape as the input $N_{sequence\,length} \cdot d_{embedding\,dimension}$, it allows to easily stack multiple encoder blocks without further modifications, thus each block can learn and attend to different relations among the inputted embedding.

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

3.1.3 Multi-Head Attention

In addition, the original Transformer model introduces a mechanism called "Multi-Head Attention" [1]. The idea of Multi-Head Attention is to train multiple attention blocks on different inputted embedding sub-spaces simultaneously in the same attention layer. This allows to capture a wider range of relationships and to encapsulate partial information from the embeddings into partial contexts. Multi-head Attention splits each embedding into h sub-embeddings before these are passed in parallel to h query W^{Q}, key W^{K} and value matrices W^{V}. In the original paper 8 heads were used [1]. Following the example from the original paper, an attention layer, which receives $N_{sequence\,length} \cdot 512_{embedding\,dimension}$ as input, splits each embedding into 8 sub-embeddings, yielding $N_{sequence\,length} \cdot 8_{heads} \cdot 64_{sub-embedding\,dimension}$, as input for the attention layer. Rather than describing this procedure mathematically with tensors, this can be thought by slicing the horizontal input sequence with it's vertical embedding vectors in 8 pieces

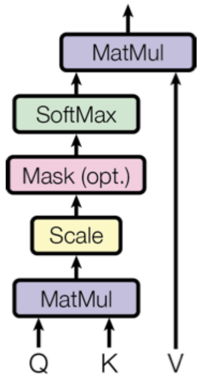

Figure 3.2: Scaled Dot-Product Attention
Source: ”Attention Is All You Need” [1]

horizontally along to obtain 8 parallel sequences, where each sequence with it’s sub-embedding vectors is separately passed to another attention block that work in parallel in the same attention layer, also referred to as ”heads”. The dimension of the sliced sub-embeddings is here described as $qkv_{dimension}$, where $h_{heads} \cdot qkv_{dimension}$ should normally yield the initial inputted embedding dimension $d_{embedding\,dimension}$. Moreover, $d_{embedding\,dimension}$ should be divisible by h_{heads} to obtain equal-sized attention blocks. 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 for multiple $\mathrm{W^Q}$, $\mathrm{W^K}$ and $\mathrm{W^V}$ matrices. After the computation within the attention block the outputted sub-embeddings are then concatenated again and multiplied by a trainable linear layer, shown in figure 3.3 as ”Linear” at the top. The multiple ”Linear” blocks at the bottom simply represent the reshaping procedure of Q, K and V into h heads. However, the ”Linear” operation at the top represent trainable weights that serve as a linear projection operation, without the use of an activation function, which weighted-based concatenate the sub-embeddings from the heads. In addition, it can ensure that the initial input shape of $N_{sequence\,length} \cdot d_{embedding\,dimension}$ is restored, even if the shape after ”Concat” does not match the initial shape. Generally, the input and the output dimension of the standard attention block has an equivalent shape of $N_{sequence\,length} \cdot d_{embedding\,dimension}$. This means that $h_{heads} \cdot qkv_{dimension}$ does not necessarily be equivalent to the embedding shape $d_{embedding\,dimension}$, thus the attention block can be of variational size and learn attention weights on a densed embedding feature space. In addition, it serves as a trainable linear projection operation, that weighted-based concatenate the sub-embeddings received from the heads and the concatenation operation. Figure 3.3 illustrates the entire Multi-Head Attention process.

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.

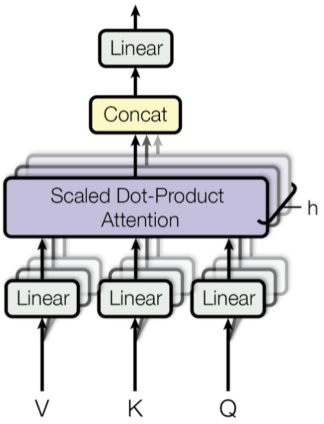

Figure 3.3: Multi-head Attention
Source: "Attention Is All You Need" [1]

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 embedding 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) [30]. 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.

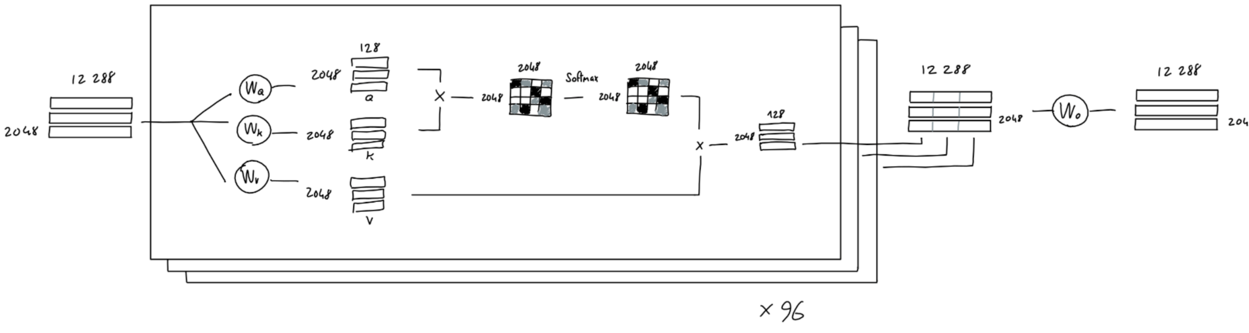

Figure 3.4: Attention mechanism & Multi-Head attention together

3.2 Mathematical background about Convolutional Networks

Convolutional networks are often applied deep-learning models for the classification of ECG signals [17] [7] [42], including the first three ranked models and many other models within the

Physionet 2021 challenge [26] [15] [43]. Traditionally used for computer vision tasks, a convolutional network can similarly be applied to one dimensional data. The main component in a convolutional network is the convolutional filter (also called kernel), which is a fixed-sized window that stripes over the input data, where each data point and it's surrounding data points within the window are multiplied with the kernel weights and added to project the data points into a new single outputted data point. The kernel then stripes over the entire input data. The outputted projection is called feature map or channel and can, depending on the configuration, have the same dimension as the input to the convolutional layer. While in the beginning the kernel weights are randomly initialized, the kernel later optimizes it's weights to the training data and can learn meaningful patterns, such as corners and edges to more abstract concepts like faces in computer vision tasks. Commonly, several convolutional filters work in parallel in the same convolutional layer, yielding several outputted features maps or channels. Therefore, a convolutional layer is typically configured by the number of convolutional filters, kernel, stride and padding size. The number of filters in a layer determines how many distinctive patterns the convolutional layer can learn within a specfic network level from the data, while each feature map contain different features from this layer. Usually, this number starts small, e.g. 32 and increases from one layer to the next deeper layer, e.g. 64, 128 and so on. The reason for this is that in deeper layers more abstract concepts need to be learned by combining simpler corner and edge features from previous layers. This process requires a larger number of filters to capture the increasing complexity and diversity of features. The filter size describes the size of the stripe window and mainly influences the number of trainable parameters. Smaller filters can focus on details, while larger filters can receipt distant associated patterns. Smaller windows may also prevent overfitting and enable deeper and more complex network hierarchies. Stride determines the number of data points the kernel window is shifted when sliding over the input data. Therefore, some data points can be left out by skipping. Padding adds margin to the borders of the input and can reduce the input size. Stride that is larger than the kernel size and padding can miss relevant input information by ignoring or skipping features, but can reduce the model size and number of trainable parameters and focus on more centered features. Beside standard convolutional filters, stride and padding, an attractive concept are dilated convolutions. Dilated convolutions employ a kernel that incorporates strides. These kernels can observe data in a different manner, extent the receptive field and reducing model size, while offering in many application still high performances. Figure 3.5 shows the idea of a dilated convolution applied on one dimensional ECG data.

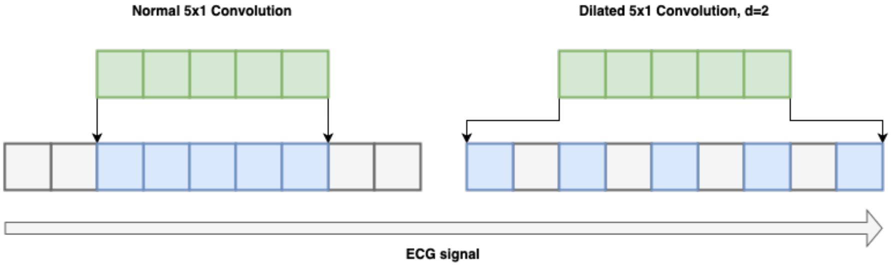

Figure 3.5: Dilated convolution

This work investigates the effect of dilated convolutions and strides for feature extraction and reducing model sizes. Typically, a convolutional layer is followed by a pooling layer that aims to reduce dimensionality. Other than the convolutional layer, the pooling layer uses a single kernel across all channels that (most commonly) applies ”min”, ”max” or ”average” pooling.

This means, the pooling kernel forwards either the minimum, the maximum value or averages the values within the window. For example, in a 1D signal a pooling window size of 2 will reduce the input dimensionality by half, because for each 2 data points a single data point is outputted and forwarded to the next layer. This process enables the convolutional network to extent the receptive field and learn from detailed features to more widely scattered features. As an alternative to pooling layers strides can be used for downsampling the input. Beside, other common components in convolutional networks include feed-forward layers using activation functions, such as sigmoid, tanh, relu, leaky relu, gelu or others. These layers can learn non-linear features, since convolutional kernels only apply linear multiplications. Commonly, the final layers of a convolutional network apply global pooling, meaning that all values within a channel are densed into a single value from which then a feed-forward or softmax layer follows as the final model output, depending on the task.

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. Another library to work with ECG data is the Neurokit2 package, which is a more maintained library, but which is due to similar methods not further discussed here. Table 3.3.1 shows 39 extracted features using the Biobss library and a short description for each feature. In the 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 these three methods implemented: "pantompkins" [28], "hamilton" [14] or "elgendi" [11]. For the experiments the "pantompkins" method is used, because according to the research work by Khan and Imtiaz [24] that compare different fiducial points detection methods, it is the most widely used approach for RR peak detection because it offers high accuracy. These methods are also the default implemented algorithms for fiducial points localisation in the Neurokit2 package. 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. After feature extraction some feature vectors had "nan", "-inf" or "inf" entries, which have been manually imputated by means. The features are then used to train several feature-based classifiers, e.g. random forest, adaboost and support vector machine. For this, the work utilises the Sklearn Python framework. The classifier 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

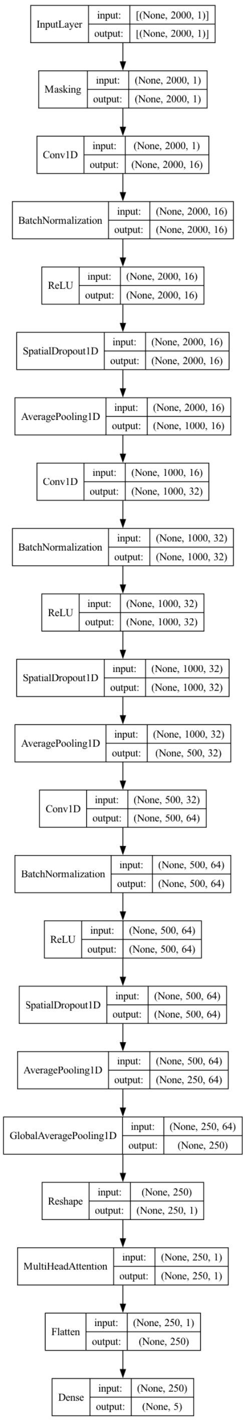

Figure 3.6: Transformer Model architecture

3.3.3 Residual CNN

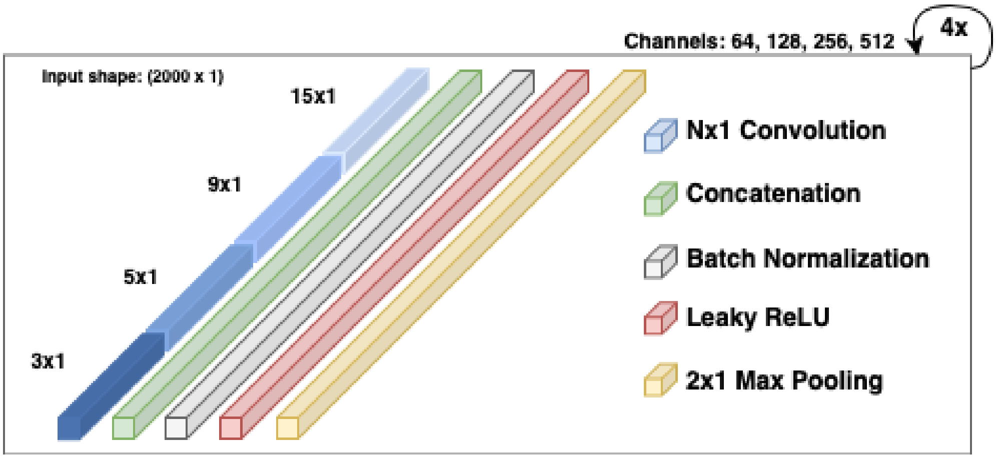

Figure 3.7: Inception Network Architecture

3.3.4 Ensembled models

3.3.5 Fine-tuning

The Physionet 2021 challenge organisiers add partial rewards to misclassifications that lead to similar cardiac treatments. Therefore the challenge score is calculated by incorporating class weights for each correct and incorrect classification. To fine-tune the performance of the models on the Physionet 2021 challenge metrics the presented approaches adopt an altered custom lost function used by the first ranked model of the Physionet 2021 challenge [26], which was initially proposed by Vicar et al. [39] [40]. The custom loss function incorporates the evaluation weights of the challenge and decreases the total loss as follows [26]:

$$\text{Loss} = \sum^{batch} (\ \text{BCE}(T,P) - \text{CL}(T,P) + \text{SL}(P)\ ) \tag{3.4}$$

$$\text{CL}\ (T,P) = \sum_{i,j=1}^{i,j=class\ n} \omega_{ij}\ a_{ij} \tag{3.5}$$

$$\text{SL}\ (P) = \text{average}(\ -4P\ (P-1)\ ) \tag{3.6}$$

The calculated total loss of the mini-batch is composed of three functions, first the standard binary cross-entropy loss for multi-label classification (BCE), second a custom loss function (CL) and third a sparsity loss function (SL). T and P are 26-dimensional vectors that contain the true labels as binary values and predicted output classes as continuous values. The sparsity loss (SL) function adds a penalty value ranging between 0 to 1 for each outputted logit that is close to 0.5. It is a down concaved parabolic curve with vertex at 0.5 and roots 0 and 1. The sparsity loss is intended to improve the final threshold optimization, as it forces the model to output values close to 0 or 1. The custom loss function incorporates the evaluation weights w_ij, where a_ij represent a multi-label confusion matrix, which use a continuous version of the logical

OR. This confusion matrix is calculated by equation A and N, where N is a normalizing constant obtained from the continuous version of the logical OR [39]:

$$A = T^T \left(\frac{P}{N} \right) \tag{3.7}$$

$$N = \left((T + P - T \odot P)\, \mathbf{1}_{c \times 1} \right)\, \mathbf{1}_{1 \times c} \tag{3.8}$$

1_{cx1} and 1_{1xc} are 26-dimensional vectors consisting of only ones, $\frac{P}{N}$ and $\odot$ are point-wise division and multiplication operators.

Chapter 4

Evaluation

4.1 Datasets

4.1.1 Physionet 2021 challenge database

This section summarises the publicly available Physionet 2021 challenge database that is used to train all models within this work. In section 4.1.2 the MyDiagnostick data provided by University of Maastricht is briefly discussed. The Physionet 2021 challenge database provides 12-lead ECGs annotated by cardiologist experts and is a diverse dataset collection from seven sources in four countries and three continents (see Physionet 2021)[31]. The sources of the challenge data include the Chapman-Shaoxing database and Ningbo database, PTB and PTB-XL database, Georgia 12-lead challenge data, CPSC and CPSC-extra database and the INCART database. In total the challenge data contains 88,253 publicly shared 12-lead ECG recordings. The annotations contain labels from more than 100 known arrhythmia types (see all Physionet 2021 database arrhythmias). Each arrhythmia class has a 'Snomed CT code' assigned, an unique ID representing each arrhythmia type. The ECGs are multi-label annotated, meaning that multiple arrhythmia types can occur within the same ECG. Table 4.1 shows an overview of the Physionet 2021 challenge database composition, the sources, recording length, sampling frequency and data proportion.

Dataset source	**Recording length in seconds**	**Sampling frequency**	**ECG samples**
Chapman-Shaoxing database & Ningbo database	10s	500 Hz	45,152
PTB database & PTB-XL database	10-120s	either 500 or 1000 Hz	22,353
Georgia 12-lead challenge database	5-10s	500 Hz	10,344
CPSC database & CPSC-extra database	6-144s	500 Hz	10,330
INCART 12-lead database	1800s (30 min)	257 Hz	74

Table 4.1: Physionet 2021 challenge data composition

This work only uses the published training data, since the test data for the official challenge and evaluation scoring metrics are withhold by the organisers and are not publicly available. Figure 4.1 shows the class distribution for the entire publicly available Physionet 2021 database (top), as well as the distribution of the scored challenge classes (bottom). The official Physionet 2021 challenge uses a subset of 30 classes that are scored within the challenge and the evaluation metrics (see scored Physionet 2021 database arrhythmias). Due to limited samples, eight arrhythmia types are treated as pairs and same class within the challenge metrics, namely Premature Atrial Contractions (PAC) and Supraventricular Premature Beats (SVPB), Premature Ventricular Contractions (PVC) and Ventricular Premature Beats (VPB), Complete Left Bundle Branch Block (CLBBB) and Left Bundle Branch Block (LBBB), Complete Right Bundle Branch Block (CRBBB) and Right Bundle Branch Block (RBBB). Therefore, to benchmark the proposed approaches, the developed models follow these requirements and annotations are similar grouped to output 26 classes instead of 30, while other class annotations and ECGs that are not part of the scored metrics are discarded. Both graphs in 4.1 show that the data is quite unbalanced. Considering the bottom graph that show the scored challenge data distribution, 28,971 (22.40%) Sinus Rhythm samples are provided, while many classes with less than 2,000 (1% <) samples are present. For an unbiased model training, the dataset needs to be pre-processed and properly balanced to avoid overfitting on some major classes, which is described in section 4.1.3.

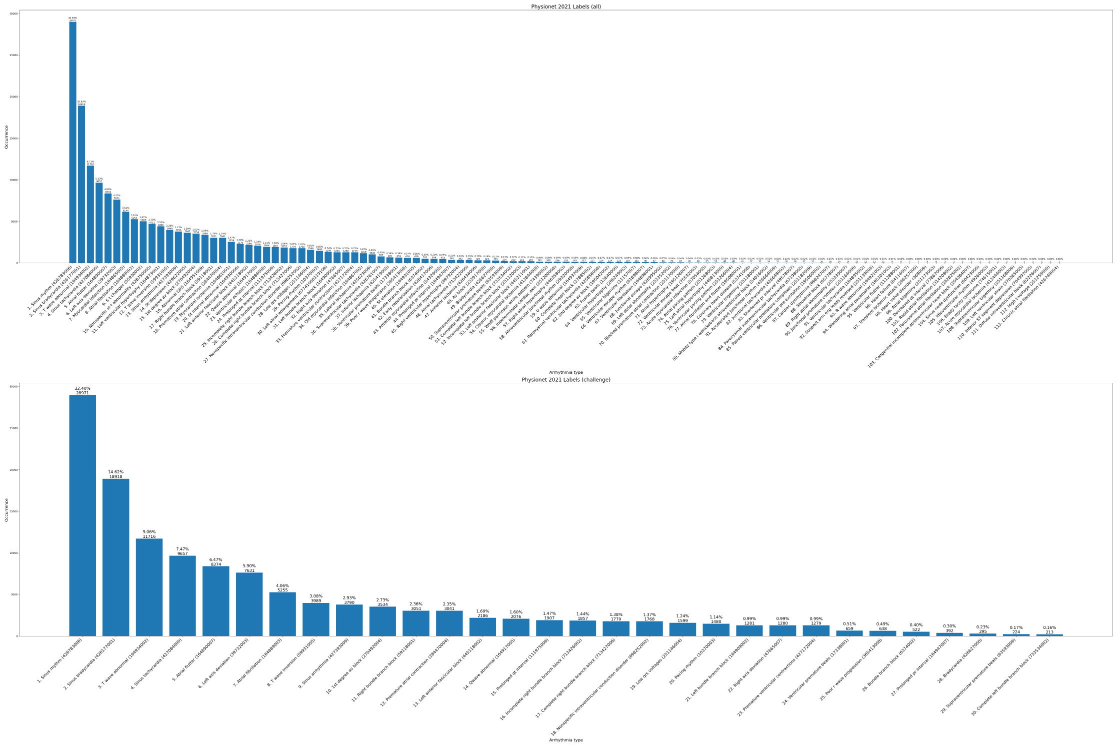

Figure 4.1: Physionet 2021 classes distribution

4.1.2 MyDiagnostick data

The MyDiagnostick data contains single-lead ECGs (first lead I) with a sampling frequency of 200 Hz and 60 seconds in recording length. For the evaluation of the transferred models on the MyDiagnostick data, part of the experiments will focus only on a subset of 5 classes, namely 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 annotations in the MyDiagnostick data, which are limited to these class annotations. ECG preprocessing steps to transfer the models are discussed in the next section 4.1.3. Figure 4.2 shows the class distribution of the corresponding classes for both, the Physionet 2021 data and the MyDiagnostick data. The most underrepresented class in both datasets is Premature Ventricular Contraction with 1279 samples, respectively with 160 samples. In many MyDiagnostick samples both classes, Atrial Ventricular Contractions and Premature Ventricular Contractions, are present within the same ECG. Although, the MyDiagnostick provides 10,587 manually annotations, the dataset consists of more than 40,000 samples. The remaining samples are automatically labelled as either Sinus Rhythm or Atrial Fibrillation by an assistive ECG device. As part of this thesis the aim of the experiments will be to evaluate the best performing pre-trained model from the Physionet 2021 challenge data on the manually annotated MyDiagnostick data and to provide annotations for the remaining 30,000 samples, which will be discussed in section 4.2.2.

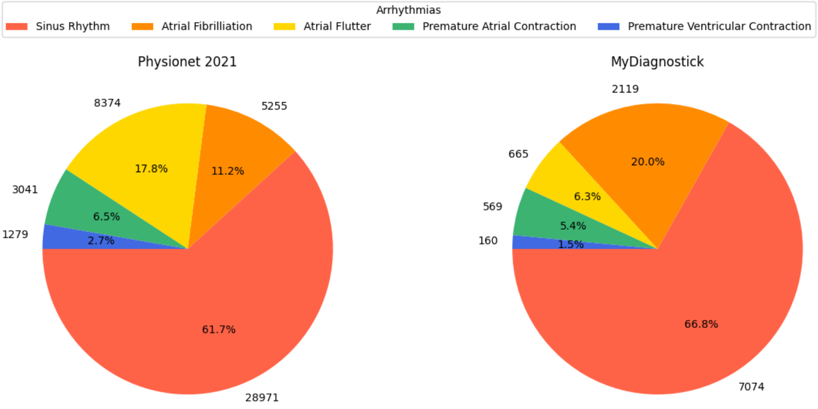

Figure 4.2: Physionet 2021 and MyDiagnostick class distribution - SR, AF, AFL, PAC and PVC

4.1.3 Pre-processing

This section summarises the pre-processing steps that have been applied on both datasets to address data inequalities. Pre-processing steps include zero padding/truncation, resampling, normalisation, filtering, train-test split by patient and class balancing. 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 data. Most of the ECGs in the Physionet data are recorded at a sampling frequency of 500 Hz, while the MyDiagnostick ECGs are recorded at a sampling frequency of 200 Hz. Most of the Physionet 2021 challenge ECGs contain 10 second long recordings (5000 data points for 10 seconds - see 4.1). In comparison the MyDiagnostick data contains 60 seconds long recorded ECGs with 12000 data points (2000 data points are equal to 10 seconds). To address the data inequality, first each ECG in the Physionet 2021 challenge is truncated or zero padded to 5000 data points. Next, each Physionet ECG is downsampled to 200 Hz, yielding 2000 data points. This makes both datasets uniform in terms of sample density and temporal resolution. In addition, each ECG is normalised within the range of -1 and 1 using min-max scaling. 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 stable training with gradient descent and avoids gradient explosion problems during backpropagation. By keeping the model weights in an uniform range prevents the model from being biased towards certain input features (e.g. R peaks). In the next step, a standard Butterworth bandpass filter is applied with a bandwidth of 0.3 Hz to 21 Hz, which reduces small noise fluctuations in the ECG signal and facilitates the model to learn key features from arrhythmias, such as RR intervals. Finally, all ECGs in the Physionet data are again zero padded to 12000 datapoints. This is necessary, because a model applied on both datasets need to accept an equal sized input feature vector. In addition, the Physionet 2021 challenge data was split by patient to ensure that the models learn general features rather than specific features of individual patients. The train-test splits are: 80% training, 20% validation and 20% test set. The MyDiagnostick data is only utilized for transferring and testing the pre-trained models from the Physionet 2021 challenge data, meaning the MyDiagnostick data is not used for fine-tuning. As in figure 3.6 shown, the training data is highly unbalanced. To address this issue, standard class down- or upsampling is applied on the Physionet data using the Python package imbalanced-learn. In the case of the 26-classes models up- and downsampling is set to 2000 samples by randomly discarding or duplicating some ECGs. In the case of the 5-classes models down- and upsampling is set to 6000 samples. The class upsampling was applied after splitting the training data by patients into train, validation and test sets to avoid evaluating on trained ECGs.

4.2 Experimental setup and results

This section describes the experimental setup and results of the developed approaches. The section is divided into two parts. The first part discusses the experimental setup and results for the models trained and tested on the Physionet 2021 database. The second part discusses the experimental setup and results of the pre-trained models on the Physionet 2021 database transferred and evaluated on the MyDiagnostick data. All models have been trained in Google Colab using an Nvidia T4 GPU with 16GB VRAM for at most 100 epochs and early stopping enabled with regard to not decreasing validation loss (patience set to 5). Adam optimizer is used with an initial learning rate of 0.001, which decreases by 1/10 with regard to not decreasing validation loss (patience set to 5). As loss function binary-crossentropy for multi-label classification is used, because the Physionet 2021 and MyDiagnostick data are both multi-label annotated. All deep-learning based models use ReLU as activation, as it offered after some prior experiments the highest performance. The model final output layers use sigmoid activation times the number of classes. For simplicity, all models are trained with a fixed mini-batch size of 32, although which could be determined empirically. Due to minimize computational costs and required training time, as each developed model took up to one hour for training, the main part of these experiments focus on a fixed train-test split, meaning that the train-test samples within these experiments are identically for each tested model. Using scikit-learn the weighted average of accuracy, precision, recall and f1 is computed for each model. In addition, the official Physionet 2021 evaluation metrics are calculated to benchmark the own developed approaches with other approaches within the challenge, see challenge results. Only the best performing model from these experiments is then also evaluated with the Physionet 20221 challenge metric using 10-fold cross-validation, yielding $10 \cdot (79,200 : 8,800)$ train and validation folds for 88,000 samples. The Physionet website provides the information that about 88,000 ECGs are shared publicly as training data and 36,266 ECGs are retained as test data. Using cross-validation and assuming that the class distribution in the official test set is similar, will produce to some extent comparable evaluation results to benchmark the developed approaches with other model results. The official Physionet challenge metrics use for evaluation AUROC, AUPRC, Accuracy, F1 measure and the own calculated challenge score.

4.2.1 Physionet 2021 challenge

This section will answer the first two research questions of this work based on the Physionet 2021 challenge data:

1. How well does a Transformer-based model perform on the Physionet 2021 challenge data compared to a feature-based model or an Convolutional Network?
2. Can an ensemble model of Transformer and Convolutional Network effectively capture spatio-temporal information and improve accuracy?

As part of the experiments to address the above research questions several variations of a standard Transformer encoder block without any feature extraction, e.g. convolution layers, are trained on the raw Physionet ECGs for the classification into 26 classes. Gridsearch is applied on various parameters. Conducted parameters include input shape, positional encoding, number of stacked encoder blocks, number of heads, dimensions of the projection matrices q, k or v, feed-forward layer dimension and dropout rate. The input shape describes in how many embeddings the inputted ECG sequence is splitted before feeded into the Transformer encoder block. (40, 50) means that the pre-processed ECG with 2000 datapoints in length is splitted into 40 temporal consecutive segments (embeddings), where each embedding contains 50 datapoints. Number of heads and q, k and v dimension describe the amount of trainable parameters of the encoder block, i.e. whether several heads are used in each encoder block and the size of the q, k and v matrices that project the embeddings into the feature space. Feed-forward dimension determines the hidden-layer dimension of the feed-forward layer that is part of each encoder block. Dropout is applied to all layers, either set 0.1 or 0.4. The corresponding results are below in table 4.2 shown.

Input shape	**Pos. enc.**	**Enc. blocks**	**Heads**	**qkv dim**	**ff dim**	**Drop-out**	**Train. param.**	**Acc.**	**Prec.**	**Rec.**	**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: Standard Transformer encoder evaluated on the Physionet 2021 challenge data

Based on the observed experimental results it can be seen that a standard Transformer encoder-based model is able to extract features from the ECGs for the classification into 26 different arrhythmias. However, in terms of accuracy there is still room for performance, while precision is in an acceptable range. Using smaller ECG segments as input sequence, e.g. (40, 50) instead of (10, 200), offers slightly improved performance. Surprising is that positional encoding negatively affects the performance. A reason for this might be that the variance of the inputted ECG sequence is to high, meaning for example that the R peaks are not equally aligned within the embeddings across the samples. By adding positional information to these varying segments might confuse or makes it harder for the multi-head attention layer to learn meaningful relations. In general the size of the model and amount of trainable parameters, e.g. number of encoder blocks, q, k and v dimension and feed-forward layer size, improves the model performance, but which should be critically seen in terms of required training costs. The effect of a higher dropout rate varies among the experiments, although in the last model it slightly improves the generalization ability of the model.

4.3 and 4.4 show the corresponding model results.

Model	Parameters	Accuracy	Precision	Recall	F-measure
Random Forest (Biobss features)	/	0.390	0.714	0.382	0.406
Residual CNN	3.702.282	0.387—0.310	0.593	0.692	0.625
Dilated CNN	0	0	0	0	0
Inception Network	53.202.522	0.358—0.299	0.581	0.673	0.609
Squeeze and Excitation Network	152.210	0.257—0.184	0.463	0.622	0.501
Multi-Head Attention	0	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)	0.560	0.142	0.390	0.153	0.147
Residual CNN	0.847	0.410	0.310	0.445	0.548
Dilated CNN	0	0	0	0	0
Inception Network	0.832	0.382	0.299	0.414	0.526
Squeeze and Excitation Network	0.768	0.290	0.184	0.322	0.445
Multi-Head Attention	0	0	0	0	0

Table 4.4: Physionet 2021 challenge metric scores model comparison

4.2.2 MyDiagnostick

Model	Accuracy	Precision	Recall	F-measure
Residual CNN 5 classes	0	0	0	0
Residual CNN 5 classes (fine-tuned)	0	0	0	0
Residual CNN 26 classes	0	0	0	0
Residual CNN 26 classes (fine-tuned)	0	0	0	0

Table 4.5: MyDiagnostick model evaluation

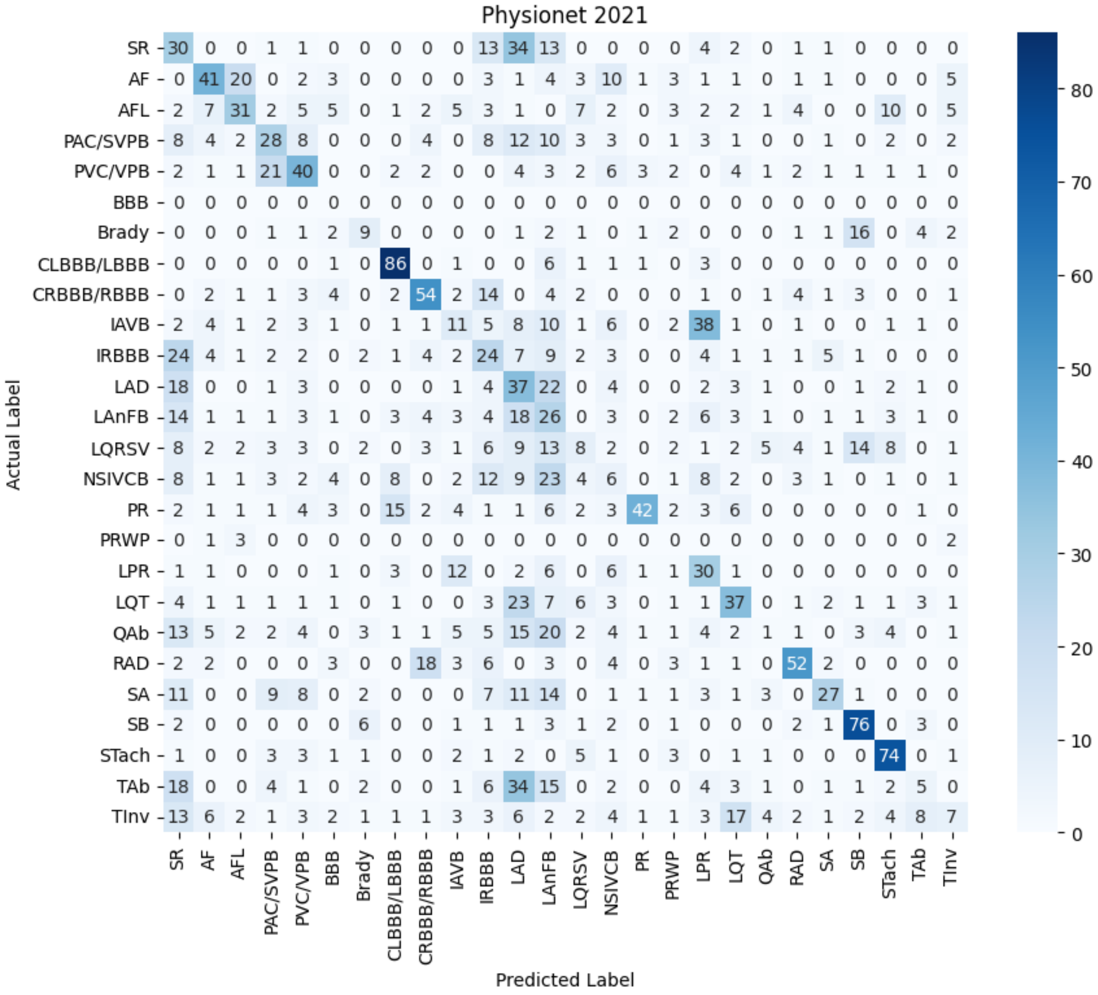

Figure 4.3: Confusion matrix - Residual CNN - scored labels

4.3 Discussion

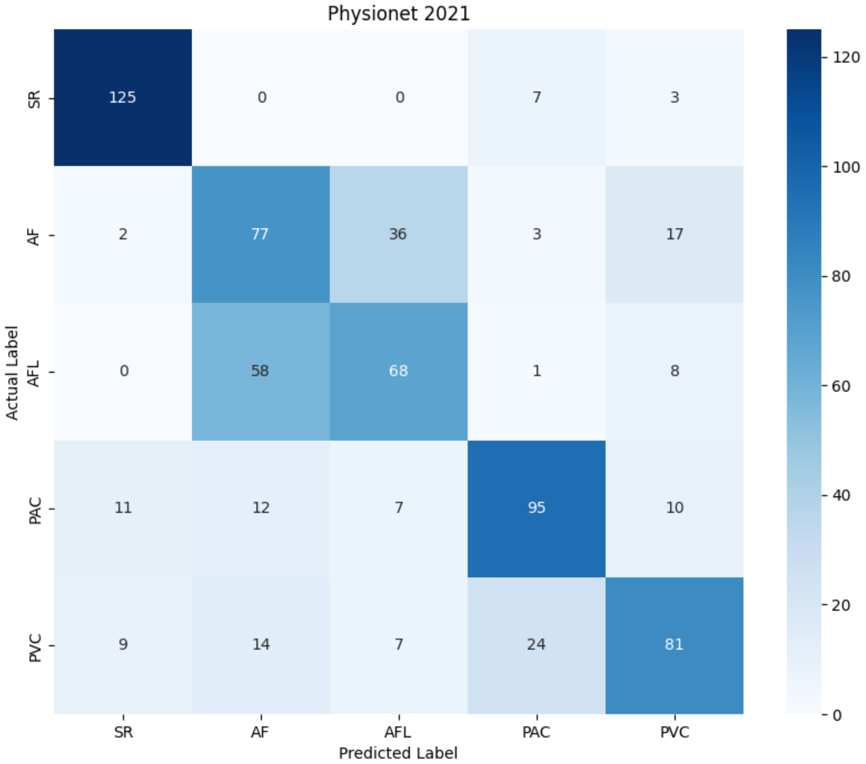

Figure 4.4: Confusion matrix - Residual CNN - SR, AF, AFL, PAC and PVC

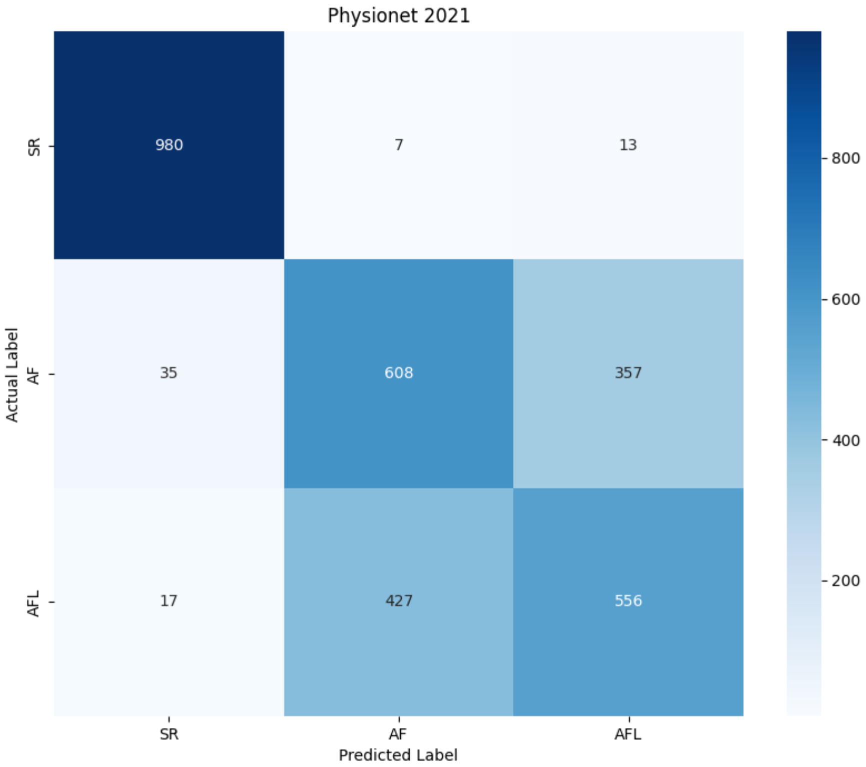

Figure 4.5: Confusion matrix - Residual CNN: SR, AF and AFL

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] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Enhua Wu. Squeeze-and-excitation networks, 2019.

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

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

[23] Maximilian Ilse, Jakub M. Tomczak, and Max Welling. Attention-based deep multiple instance learning, 2018.

[24] Naimul Khan and Md Niaz Imtiaz. Pan-tompkins++: A robust approach to detect r-peaks in ecg signals, 2022.

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

[26] 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.

[27] 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.

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

[29] 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.

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

[31] 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.

[32] 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.

[33] 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.

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

[35] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge, 2015.

[36] 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.

[37] Apoorva Srivastava, Ajith Hari, Sawon Pratiher, Sazedul Alam, Nirmalya Ghosh, Nilanjan Banerjee, and Amit Patra. Channel self-attention deep learning framework for multi-cardiac abnormality diagnosis from varied-lead ecg signals. 2021.

[38] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. 2014.

[39] Tomas Vicar, Jakub Hejc, Petra Novotna, Marina Ronzhina, and Oto Janoušek. Ecg abnormalities recognition using convolutional network with global skip connections and custom loss function. 2020.

[40] Tomas Vicar, Petra Novotna, Jakub Hejc, Oto Janousek, and Marina Ronzhina. Cardiac abnormalities recognition in ecg using a convolutional network with attention and input with an adaptable number of leads. 2021.

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

[42] 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.

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

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

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

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

[47] 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.